https://ntrs.nasa.gov/search.jsp?R=19660014479 2020-03-16T19:43:05+00:00Z

This Research Was Sponsored by

THE NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Design of Countup-Countdown Machines

Report No. EDC 1-65-31

by

George J. Moshos

Harry W. Mergler
Professor of Engineering
Principal Investigator
Ns G 36-60

Digital
Systems
Laboratory

1965

ABSTRACT

23968

The design of a class of special purpose computing machines which compute by counting is systematically developed. The basis of the design philosophy is to limit the basic building elements to three fundamental units and to develop the method of synthesis such that these three building elements are represented as operational units. In particular, the three basic building elements are (1) the binary rate multiplier which is a means of scaling down a pulse stream to some specified fraction, (2) the counter, and (3) the anti-coincidence circuit which is a means of separating pulses arriving at the counter simultaneously. The computational errors; i.e., rounding-off error and truncation error, introduced into the machines when these elements are treated as operational units are studied in detail. The method of synthesis is explicitly stated and a wide variety of machines obtained directly from this synthesis are presented. Finally, a series of machines is presented for interpolation and extrapolation of a function which is available only as empirical data.

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER I

PRINCIPLE DESIGN ELEMENTS

Introduction

Computers are usually divided into two broad catagories, analog and digital. Analog computers represent variables as physical quantities. The solution of a problem in an analog computer is attained by constraining a physical model of the problem to be solved, and measuring the variables. The ability to program a wide variety of problems is achieved by having functional components available (e.g., adders, multipliers, integrators) and interconnecting them by means of a patchboard. The resulting interconnection is scaled to match the desired equation. On the other hand, digital computers represent variables as discrete quantities. The usual method of solution of a problem in a digital computer is attained by sequencing a sequence of instructions through the fetch-execute cycle of its control unit. Another class of digital computers, known as incremental computers, combine the parallel functional simplicity and speed of analog computers with the ability of attaining computational precision which is not dependent on precision of measurements. Such computers attain a speed advantage over conventional general purpose computers by transmitting and processing only partial words in a number of parallel arithmetic organs rather than the whole words needed by the fetch-execute cycle. Moreover the digital nature of these computers permit the problem solution to be repeated exactly and therefore does not possess the drift characteristic of analog computers. Beyond a doubt the incremental computer which has found the most interest in the literature is the digital differential analyzer; i.e., DDA. This computer can be viewed as a digital analogy of an analog computer. The usual design practice in each of these machines is to permit them to solve a large spectrum of problems. When a computer need arises for a special purpose application, this versatility is felt as a cost factor.

A class of incremental techniques which has been used in real time control is a class known in the industry as countup-countdown techniques. The basis of these techniques is to represent data by a unitary code. For example, the number 28 is represented by 28 pulses. A function may be represented by counting the sequence of pulses in a forward-backward counter or converting them directly into an analog quantity (e.g., by a stepping motor) for analog processing. Consequently, when a real time application deals with continuous-smoothly varying functions, countup-countdown techniques offer a simplicity and economy of hardware which is hard to beat with computing systems designed to handle a large spectrum of problems.

The purpose of this thesis is to investigate countup-countdown techniques with the objective of demonstrating that they can, in fact, be used to generate a wide variety of non-trival functions. This will be done by displaying a circuit which will generate each function. However, since the techniques upon which we base this thesis are described in the literature only in an ad-hoc manner

(Refs. 4, 5, 9, 10, 12, and 19), we will be specific as to which circuits we will permit as basic building elements. In particular, the fundamental units which we will permit are (1) the binary rate multiplier (abbreviated BRM) which is a means of scaling down a pulse stream to some specified fraction, (2) the counter, and (3) the anticoincidence circuit which is a means of separating pulses arriving at the counter simultaneously. In order to strengthen our argument we will avoid completely the explicit use of adders and subtractors. A succinct recapitulation of the purpose of this study is to systematically develop and demonstrate the versatility of techniques based on counting for solving sophisticated and practical special purpose computer design problems.

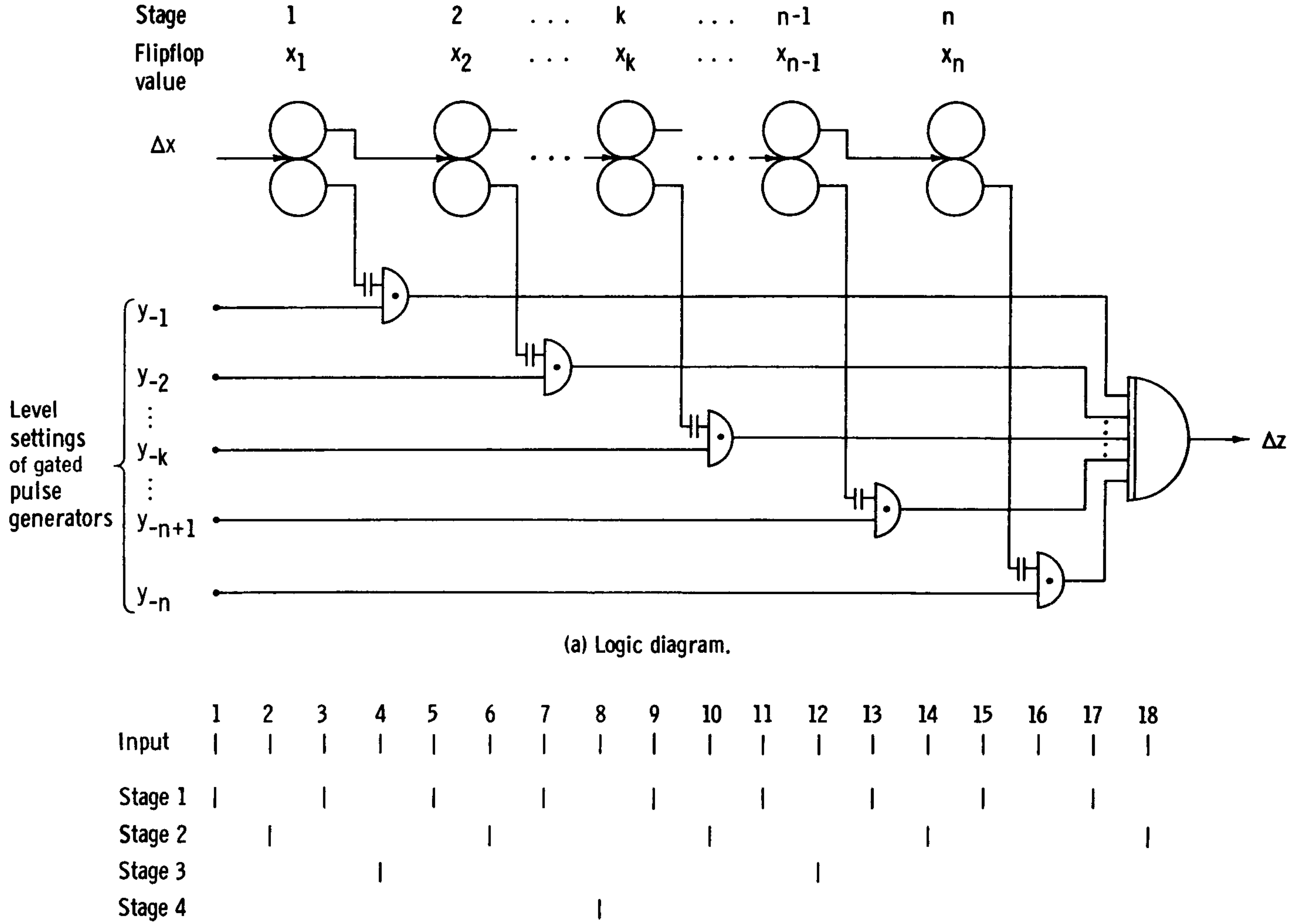
Our method of synthesis will be to describe the principle building elements as operational units and then proceed by operational techniques to show how to fabricate the various machines. In particular, a first order difference equation can be represented by a counter, and approximate integration can be attained by using a counter in cascade with a binary rate multiplier. These principle design elements are described in this chapter.

It is to be expected that the results obtained by operational means will deviate from the actual results due to the finiteness of the machine and the approximation implied by our synthesis. A discussion of these approximations is presented in CHAPTER II. This chapter is supplemented by Appendices A and B where some quantitative results are presented related to the computational accuracy of the BRM. In CHAPTER III we explicitly state the method of synthesis and demonstrate it by deriving a wide variety of representative machines. Some of these machines have been simulated on a general purpose computer and these results are also presented and discussed in CHAPTER III. In CHAPTER IV the specific problems of constructing polynomial generating machines are considered. In particular, a family of machines are given for interpolating and extrapolating values of a function defined only by empirical data.

Binary Rate Multiplier

A binary rate multiplier (abbreviated BRM) is a means of scaling down a pulse stream to some specified fraction. A logic diagram of a BRM which is built out of standard logic elements is shown in Fig. 1.1a. This circuit is described in detail in several of the references (e.g., Refs. 4 and 10). Consequently, a brief description will serve our purposes. The input pulse stream is applied directly to the binary counter whose value is denoted by $x_n x_{n-1} \cdot \cdot \cdot x_2 x_1$. Each flip-flop of the counter is operated as a trigger. For every two input pulses to a trigger two output pulses are produced; one pulse when the flip-flop makes a 0 to 1 transition called an α pulse and one when the flip-flop makes a 1 to 0 transition called a β pulse. The β pulse is used to trigger the successive stage of the counter. The α pulses are gated through gated pulse generators and mixed through a NOR element to produce the desired fraction of the input pulses. This simple mixing technique may be used because the α pulses from the various stages are separated in time from each other. This timing factor is shown in Fig. 1.1b.

The quantitative relationship of a BRM may be expressed as



(a) Logic diagram.

(b) Timing diagram.

Figure 1.1. - Binary rate multiplier.

follows: If Δx is the number of input pulses, the number of output pulses produced by the k^{th} stage of the counter is $\Delta x \cdot 2^{-k}$. This multiplicative relation will remain valid over any interval for which Δx is a multiple of 2^k pulses. If y_{-k} is the level setting of the k^{th} stage gated pulse generator, the number of output pulses which may be gated through this stage will be $y_{-k} \Delta x \cdot 2^{-k}$. Since the output pulses from the various stages are simply mixed, the number of output pulses Δz of an n stage BRM over any interval Δx which is a multiple of 2^n pulses will be the sum of all the pulses gated through all the stages. This output is

$$\Delta z = \Delta x \sum_{i=1}^{n} y_{-i} 2^{-i} \tag{1.1}$$

The quantity $y = \sum_{i=1}^{n} y_{-i} 2^{-i}$ is a binary number. Therefore Eq. (1.1) may be written as

$$\Delta z = y \, \Delta x \tag{1.2}$$

where the range of y is

$$0 \leq y \leq 1 - 2^{-n} \text{ in steps of } 2^{-n} \tag{1.3}$$

If y remains constant over a Δx interval of 2^n pulses, then the output shown by Eq. (1.2) remains exact. However if Δx is less than 2^n pulses then this multiplicative relationship remains valid only on the average. This can be demonstrated as follows: If Δx is the number of input pulses into a n-stage BRM starting with counter value x, and whose gated pulse generators are set to value y, then the output for this machine is Δz_x. Since there are 2^n possible starting values, then there are 2^n possible different machines. The average output; denote it by $\overline{\Delta z}$, over all of these 2^n different machines is

$$2^n \, \overline{\Delta z} = \sum_{x=0}^{2^n-1} \Delta z_x \tag{1.4}$$

The total pulse output over all of these machines is $\sum_{x=0}^{2^n-1} \Delta z_x$ pulses. This is equivalent to putting $2^n \, \Delta x$ successive input pulses into a single machine since each transition over all of these machines is attained Δx times. For example, the transition ending with counter value x is attained by the Δx machines starting out with the counter value prior to x. Therefore, the total number of output pulses over all of these machines is also given by Eq. (1.2).

$$\sum_{x=0}^{2^n-1} \Delta z_x = y \cdot 2^n \, \Delta x \tag{1.5}$$

Combining Eqs. (1.4) and (1.5) we have

$$\overline{\Delta z} = y \, \Delta x \tag{1.6}$$

Because of the approximate nature of Eq. (1.2) when Δx is less than 2^n pulses, we will calculate the specific output sequence in demonstrating specific machines. For these calculations, the pulse stream shown in Fig. 1.1b may be displayed in vector form. This will be called the p-sequence. Each position of the vector in this sequence represents the possible output at a particular pulse time from a stage of the BRM. The p-sequence for a two, three, and four stage BRM is displayed in Table 1.1.

TABLE 1.1 - EXAMPLES OF p-SEQUENCES

Pulse	2 Stage p-sequence	3 Stage p-sequence	4 Stage p-sequence
1	10	100	1000
2	01	010	0100
3	10	100	1000
4	00	001	0010
5		100	1000
6		010	0100
7		100	1000
8		000	0001
9			1000
10			0100
11			1000
12			0010
13			1000
14			0100
15			1000
16			0000

The p-sequences given above assumes that the BRM counter starting value is zero. If another starting value is used then its associated p-sequence can be easily obtained. Moreover, if an interval greater than 2^n pulses is used, then the p-sequence can be obtained by repeating the p-sequence given above.

The sequence of output pulses may be calculated by multiplying bit-by-bit the p-sequence with the respective values of the level settings of the gated pulse generators. This process is illustrated below by two examples.

Example A:

$$\begin{pmatrix} 100 \\ 010 \\ 100 \\ 001 \\ 100 \\ 010 \\ 100 \\ 000 \end{pmatrix} \begin{pmatrix} 01010101 \\ 10111011 \\ 11101111 \end{pmatrix} \doteq \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Example B:

$$\begin{pmatrix} 100 \\ 010 \\ 100 \\ 001 \\ 100 \\ 010 \\ 100 \end{pmatrix} \begin{pmatrix} 1010101 \\ 0100010 \\ 0001000 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

The first matrix, in each of these examples, is the p-sequence. The next matrix represents successive values of the gate settings. When these two matrices are multiplied, the result is developed along the diagonal of the resultant matrix. This result is shown as a vector on the right hand side.

The expected output value of Example A by Eq. (1.6) is 35/8 pulses for the 8 input pulses. However, as shown by actual computation, the BRM yields zero output pulses. On the other hand, the expected value of Example B by Eq. (1.6) is 21/8 pulses for the 7 input pulses. The above computation yields 7 output pulses. Both of these examples are pathological cases in the use of the BRM. The approximate nature of Eq. (1.6) can ordinarily be expected to yield more realistic results. Some of these results are presented in CHAPTER III.

The method of synthesis to be presented necessitates that the BRM operate on signed quantities. In particular, the level setting of

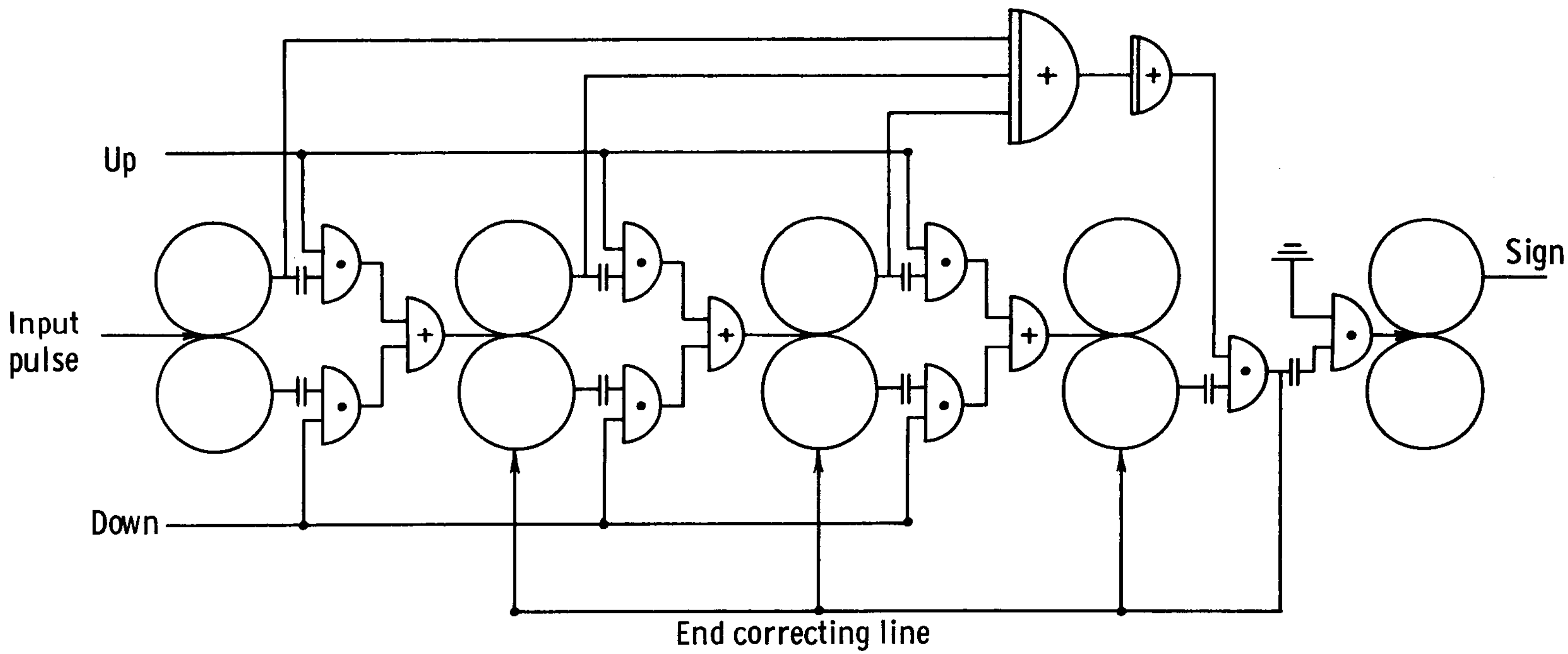


Figure 1.2. - Signed four-stage bidirectional counter.

the gated pulse generators and the counter input pulses must be signed quantities, and the BRM is to yield a signed pulse output. If the output pulses are accumulated in a counter then the sign of the pulse will determine the direction of counting. If the output pulses are used to drive a stepping motor, then the sign of the pulse will determine the direction the stepping motor is to turn. Throughout this discussion we will consider that the signs of various quantities are available through level logic. Consequently, the output sign can be obtained from the input signs by an exclusive OR circuit.

Counter

The purpose of the counter in the machines which will be considered are twofold; (1) to accumulate the pulses arriving at the counter in order to display the total number of counts, and (2) to set the levels of the BRM's. In the first application the counting sequence can be any desired sequence for a terminal device. In many real time applications the output pulses may not be accumulated directly but are converted to an analog quantity for analog processing (e.g., by a stepping motor). In the second application, the counting sequence must be compatible with the BRM. This general requirement can be met by the circuit displayed in Fig. 1.2.

A number is represented in this counter by magnitude plus sign. As had been stated earlier the signs are represented by level logic. The counter counts down in magnitude when the input pulse and counter are opposite in sign and counts up in magnitude when the counter and input pulse have the same sign. The circuit is designed so that the α pulses are used to count down and the β pulses used to count up.

There are two representations of zero; that is, minus zero and positive zero. When the counter value is at +00 . . . 0 and a -1 pulse arrives then the counter is set to -00 . . . 01. This end correction is accomplished in three steps. The normal sequence first changes the counter value to +11 . . . 1. The magnitude is then corrected in the second step to +0 . . . 01. Finally, the sign is changed to -00 . . . 01. The sign is changed last so that the β pulses generated when the magnitude is corrected do not propagate to the successive stages of the counter. In a similar manner to that just given, the counter is set to +00 . . . 01 when the counter values is -00 . . . 0 and a +1 pulse arrives. The down counting sequence for a three stage counter is given in Table 1.2.

TABLE 1.2 - DOWN COUNTING SEQUENCE

-1 Input pulse	+1 Input pulse
+111	-111
+110	-110
+101	-101
+100	-100
+011	-011
+010	-010
+001	-001
+000	-000
+111 → +001 → -001	-111 → -001 → +001

The up counting sequence utilizing the β pulses of the flip-flops is given in Table 1.3.

TABLE 1.3 - UP COUNTING SEQUENCE

-1 Input pulse	+1 Input pulse
-000	+000
-001	+001
-010	+010
-011	+011
-100	+100
-101	+101
-110	+110
-111	+111

Since the signs of both the pulse output of the BRM and counter value are to be processed by level logic, then the activation of the up-down line is accomplished by an exclusive OR circuit. This is obvious from Table 1.4.

TABLE 1.4 - COUNTER SIGN CONTROL

Sign input pulse	Sign counter	Line activated
+	+	Up
+	-	Down
-	+	Down
-	-	Up

Anti-Coincidence Circuit

Pulses arriving at a counter simultaneously must first be separated before they are entered into the counter. The circuit that accomplishes this task is called an anti-coincidence circuit. Fundamentally, this circuit necessitates storing each pulse as it arrives. Each stored pulse is then presented to the counter according to a fixed program. The circuit configuration which can accomplish this task for two inputs is shown in Fig. 1.3.

The operation of the circuit given in Fig. 1.3 is as follows:

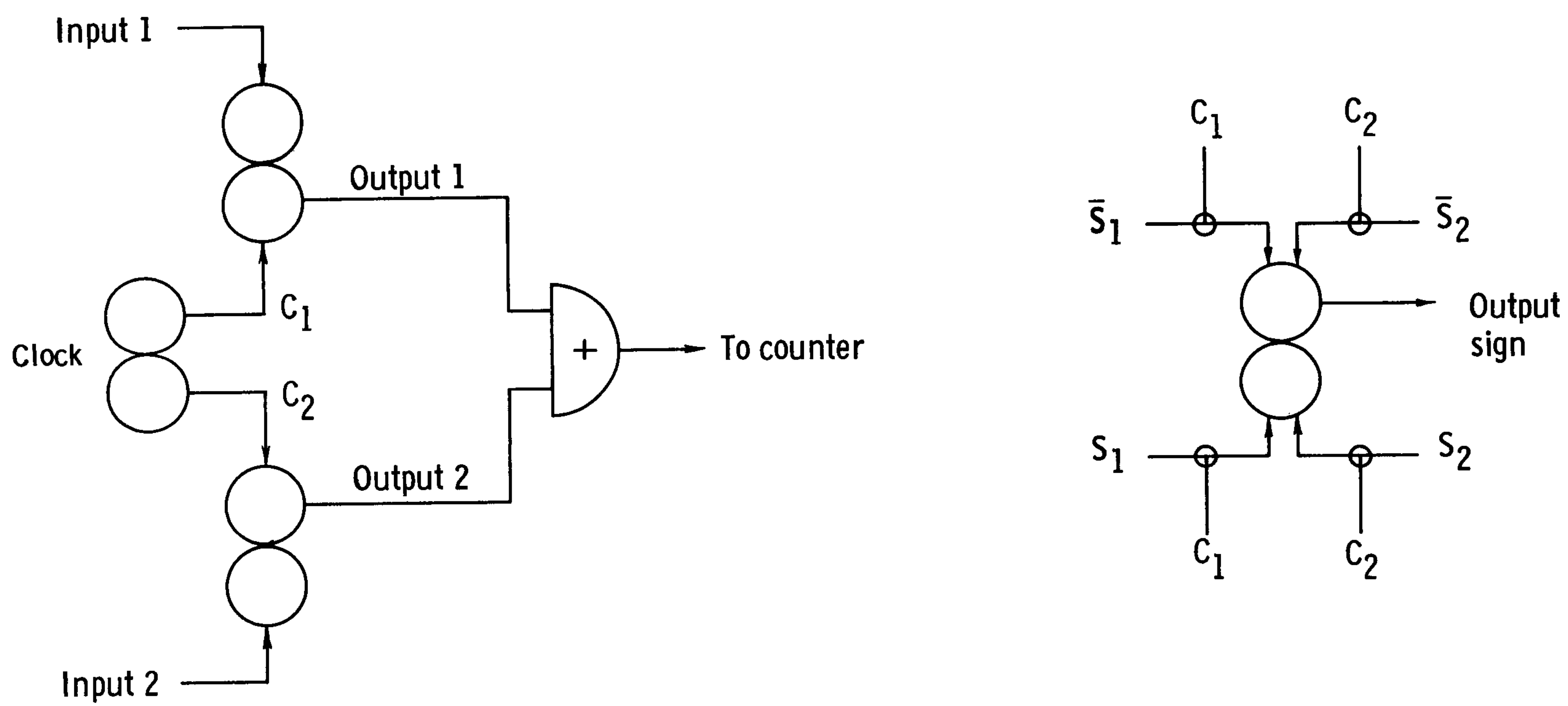


Figure 1.3. - Anti-coincidence circuit.

If a pulse from input 1 exists, it is stored in flip-flop 1. If a pulse from input 2 exists, it is stored in flip-flop 2. It will be noted that these two inputs can arrive simultaneously. Two pulses are emitted by the clock which are separated from each other. If flip-flop 1 had been set by input 1, it is reset by the clock pulse C_1, which in turn generates an output pulse. If flip-flop 1 had not been set, no output pulse will appear in the output. Flip-flop 2 is reset and an output is similarly generated by clock pulse C_2. Since clock pulses C_1 and C_2 are separated, the corresponding output pulses are also separated.

Since the sign of a pulse is processed by level logic, the sign need not be stored before they are presented to the anti-coincidence circuit. However, when a pulse is presented to the counter, its sign must also be presented. This may be simply accomplished by shifting the sign level to a flip-flop by the separated clock pulses C_1 and C_2. This circuit is also shown in Fig. 1.3 where S's and $\bar{S}$'s are the sign levels of the pulses and their complements, respectively.

If more than two inputs arrive at the counter simultaneously, then a need arises for a circuit other than a simple clock to separate the stored pulses A simple binary counting sequence such as the leftmost sequence shown in Table 1.5 will serve this purpose. However, it will be noted that while this sequence can generate more than two steps, the α and β pulses from the various flip-flops are not separated, so consequently can not both be used.

All the sides of the flip-flops could be used if the counting sequence utilizes a unit distance code. Such a code would guarantee that not more than one flip-flop would change state at any step of the counting sequence. Consider the Gray code counting sequence given by the middle sequence in Table 1.5. This counting sequence can be used to separate as many as six inputs arriving at the counter simultaneously. However, this requires the counter itself to go through eight steps. An example of a counting sequence which may be used to handle six inputs and yet go through only six steps in the counting sequence is given by the rightmost sequence in Table 1.5.

TABLE 1.5 - PULSES GENERATED BY SEVERAL COUNTING SEQUENCES

Counting sequence	Pulses generated	Counting sequence	Pulses generated	Counting sequence	Pulses generated
000	--α	000	--α	000	--α
001	-αβ	001	-α-	001	-α-
010	--α	011	--β	011	α--
011	αββ	010	α--	111	--β
100	--α	110	--α	110	-β-
101	-αβ	111	-β-	100	β--
110	--α	101	--β		
111	βββ	100	β--		

Schematic Representation

The three circuits described in this chapter are the principle design elements. However, in describing the machines promised by this thesis, these three circuits will be represented as operational units. The advantages to be gained by using operational units rather than these circuits are twofold. First, the method of synthesis can be more clearly presented. Secondly, a considerable hardware reduction can usually be realized when the composite machine is considered. These simplifications arise when all of the features of these basic circuits are not required. A checklist of the features which when removed would simplify the basic circuits would include: (1) sign control of BRM may not be required, (2) counter may not be required to both countup and countdown, (3) two BRM's may receive the same input pulse stream with the result that one BRM counter may be used with two sets of gated pulse generators, and (4) the level setting of the BRM may be constant with the result that a scaling circuit (see Ref. 10) rather than a BRM may be used. These three circuits are, however, sufficient as principle design elements.

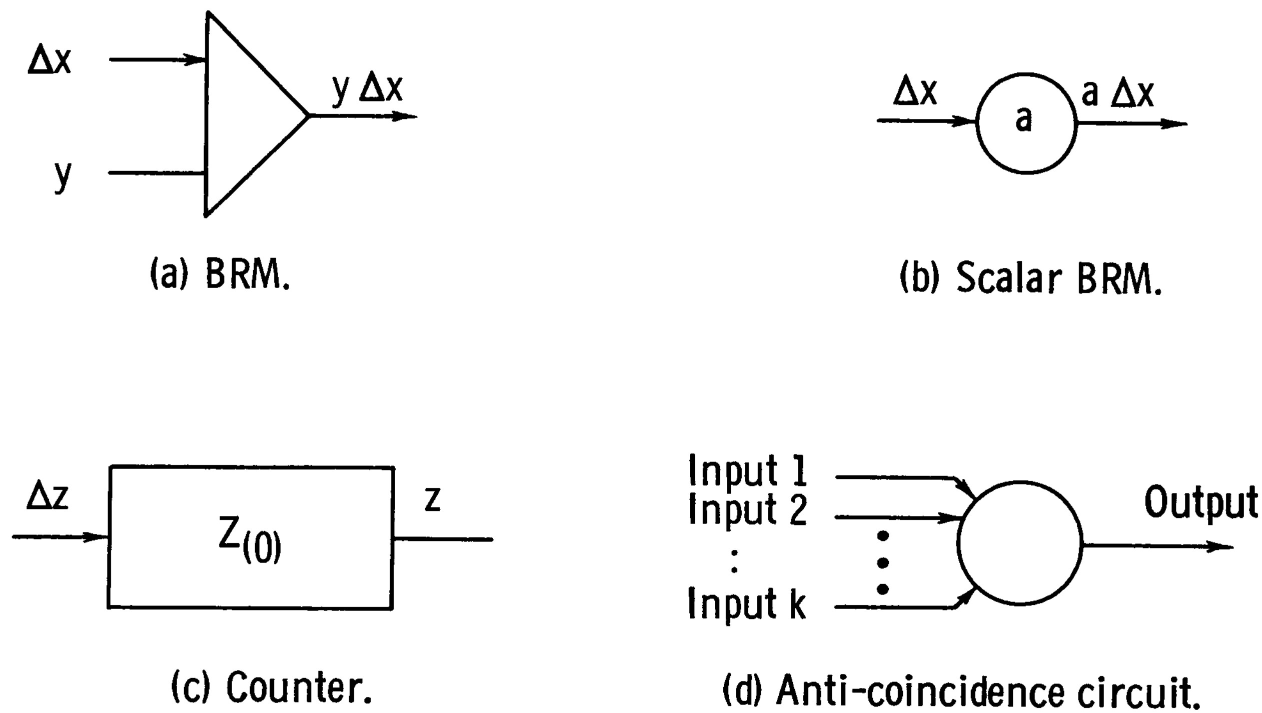
The three principle design elements as operational units are presented in Fig. 1.4. The BRM is represented by the schematic diagram shown in Fig. 1.4a. The value y in this diagram is less than one and is obtained from level logic. The quantities Δx and Δz are the input and output pulse streams, respectively. The input-output relation for this diagram is expressed by Eq. (1.2). Alternately, the BRM is represented by the schematic diagram shown in Fig. 1.4b when the value of y remains constant. In these cases the BRM may be replaced by a scaling circuit in the final design. Fig. 1.4c presents the schematic diagram of the counter. The quantity Δz is an input pulse stream and z is the output which may be used in level logic. When the counter is used to set the levels of the gated pulse generators of a BRM a scale reduction of 2^{-n} is implied by the connection. At times this scale reduction will be shown explicitly by the same diagram shown in Fig. 1.4b. The initial conditions of a counter may be shown explicitly by inserting it in the box; i.e., $z_{(o)}$. Considering the counter value as a function of iterative steps then the input-output relation may be expressed by the first order difference equation

$$z_{(k)} = z_{(k-1)} + \Delta z_{(k-1)} \tag{1.7}$$

The value of $z_{(k)}$ in Eq. (1.7) in terms of the initial condition of the counter is

$$z_{(k)} = z_{(o)} + \sum_{i=0}^{k-1} \Delta z_{(i)} \tag{1.8}$$

Fig. 1.4d represents the schematic diagram of an anti-coincidence circuit. This circuit accepts multiple pulse inputs and produces a single pulse output. The design of this circuit is such as to permit the input pulses to arrive simultaneously. However, at times it will be convenient to use this schematic diagram for multiple pulse inputs even if the pulses are known to be separated. This usage, therefore, should permit a corresponding simplification in the final design.



(a) BRM.

(b) Scalar BRM.

(c) Counter.

(d) Anti-coincidence circuit.

Figure 1.4. - Schematic diagrams of principle design elements.

CHAPTER II

ANALYSIS OF MACHINE COMPUTATIONAL ERRORS

Classification of Errors

The difference between the actual output of a system and that given by a theoretical model is considered the error of the system. If f represents the theoretical process given by the model, x represents the values of the arguments, f_a represents the actual process, and x_a represents the values of the arguments vitiated by previous calculations, then the total error $\mathcal{E}$ is given by

$$\mathcal{E} = f(x) - f_a(x_a) \tag{2.1}$$

It is convenient to subdivide the error into the error propagated from previous calculations and error generated locally. The sum of these two errors is also equal to the total error as is evidenced by rewriting Eq. (2.1) as:

$$\mathcal{E} = f(x) - f(x_a) + f(x_a) - f_a(x_a) \tag{2.2}$$

The quantity $f(x) - f(x_a)$ is error propagated from previous calculations and is called the propagated error. The difference between the value calculated locally by the model and the value generated by the actual process; i.e., $f(x_a) - f_a(x_a)$, is called the generated error.

von Neuman and Goldstine (Ref. 1) classified the generated errors into four categories according to their source. These four sources are simply listed as follows:

(1) Model errors

(2) Input errors

(3) Truncation errors

(4) Rounding-off errors

The last two errors are of primary concern to the numerical analyst and are called computational errors. Truncation errors result from expressing transcendental operations as numerical processes. For example, if a transcendental function is evaluated by an infinite series or as the fixed point of a process, then the truncation error is the error introduced by terminating the evaluation short of the limit goals. In the case of an iterative process, this error is called the iterative truncation error. Rounding-off error is introduced into the resultant after each arithmetic operation. In conventional digital computers, this error is introduced by rounding or chopping a number such that it can be represented by a register of fixed length. It can be viewed as an error in the arithmetic processes.

If f_c represents the numerical approximation of the theoretical process f, then the difference $f(x_a) - f_c(x_a)$ is the generated truncation error and $f_c(x_a) - f_a(x_a)$ is the generated rounding error. The generated error is equal to the sum of these two errors as is evidenced by

$$f(x_a) - f_a(x_a) = f(x_a) - f_c(x_a) + f_c(x_a) - f_a(x_a) \quad (2.3)$$

Computational Errors of BRM

Fig. 2.1 represents a counter in cascade with a BRM. If $y_{(k)}$ in this figure represents successive values of y, each of which remain constant over some interval of 2^n pulses, then the value of the counter can be expressed by the following difference equation.

$$z_{(k)} = z_{(k-1)} + y_{(k-1)} \Delta x \qquad (2.4)$$

If $z_{(0)}$ represents the initial value of the z counter, then Eq. (2.4) may be expressed as

$$z_{(k)} = z_{(0)} + \sum_{i=0}^{k-1} y_{(i)} \Delta x \qquad (2.5)$$

This equation is recognized as Euler's (rectangular) integration.

A model of this process which is convenient for machine synthesis is presented in Fig. 2.2. The deviation of the results given by the model from that given by Eq. (2.4) is the truncation error.

The counter z in Fig. 2.1 may be viewed as a lower register z_l consisting of n stages and an upper register z_u of an arbitrary number of stages. Thus, after the first iteration of Eq. (2.4), z_l contains the fraction $y_{(0)}\, 2^n$ of input pulses. Rounding of the upper register may be accomplished by presetting z_l to one-half of the maximum counts possible in z_l, and chopping may be accomplished by presetting z_l to zero. After the second iteration, $y_{(1)}\, 2^n$ pulses are added to the counter z. As a result of this iteration, z_l may or may not overflow into z_u. Proceeding in this manner, it is noted that z_u represents the single precision rounded or chopped sum shown in Eq. (2.5).

The above description has been presented only in order to put

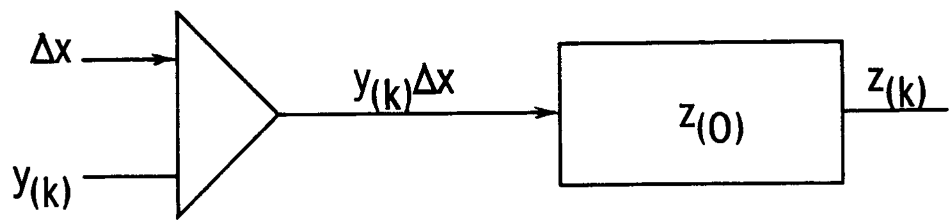


Figure 2.1. - Euler's integration.

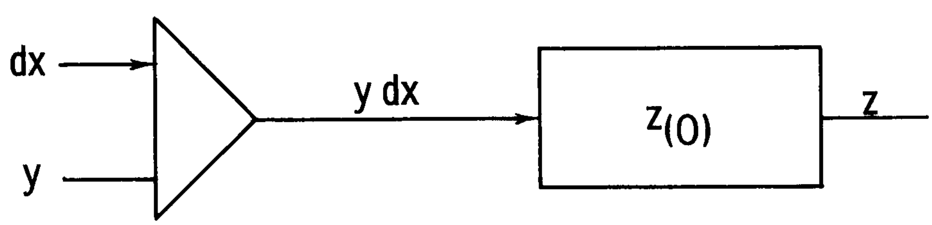


Figure 2.2. - Integration model.

the rounding error in the framework of conventional computers. A more realistic operational procedure may be realized is Δx is a subinterval of less than 2^n pulses. The advantages to be gained in this situation are twofold; (1) the speed of the computation may be increased, and (2) a hardware savings may be realized by reducing or eliminating the lower register. The output in this case is vitiated also by an error in multiplication. In particular, if Δx is a subinterval of one pulse then the output is in error only by the error in multiplication. Because of the importance of Eq. (2.4) in this study, the multiplication error of this equation is presented in Appendices A and B and in the following sections.

Multiplication Error Formulas

Starting the BRM counter with zero the actual output of the BRM is given by

$$z = \text{Entier}\ (y\ \Delta x + 1/2) \tag{2.6}$$

This function together with a plot of Eq. (1.2) is given in Fig. (2.3) for a three stage BRM for the various values of y. The difference E between these two quantities; i.e.,

$$E = \text{Entier}\ (y\ \Delta x + 1/2) - y\ \Delta x \tag{2.7}$$

is plotted for this three stage BRM in Fig. (2.4). The difference E, when only one stage of an n stage BRM is gated by y, can also be expressed systematically in tabular form as shown in Table 2.1.

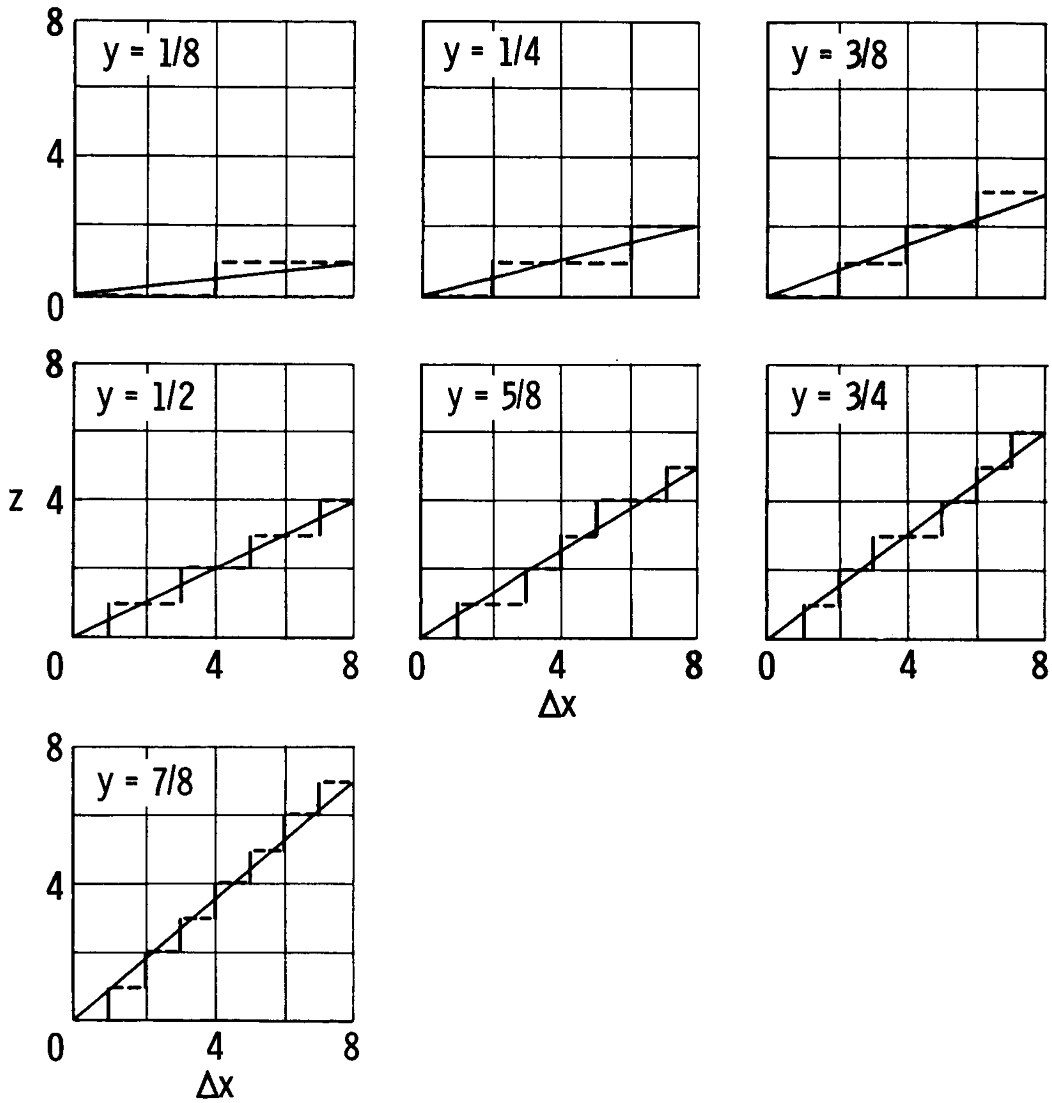


Figure 2.3. - Three stage BRM desired and actual outputs.

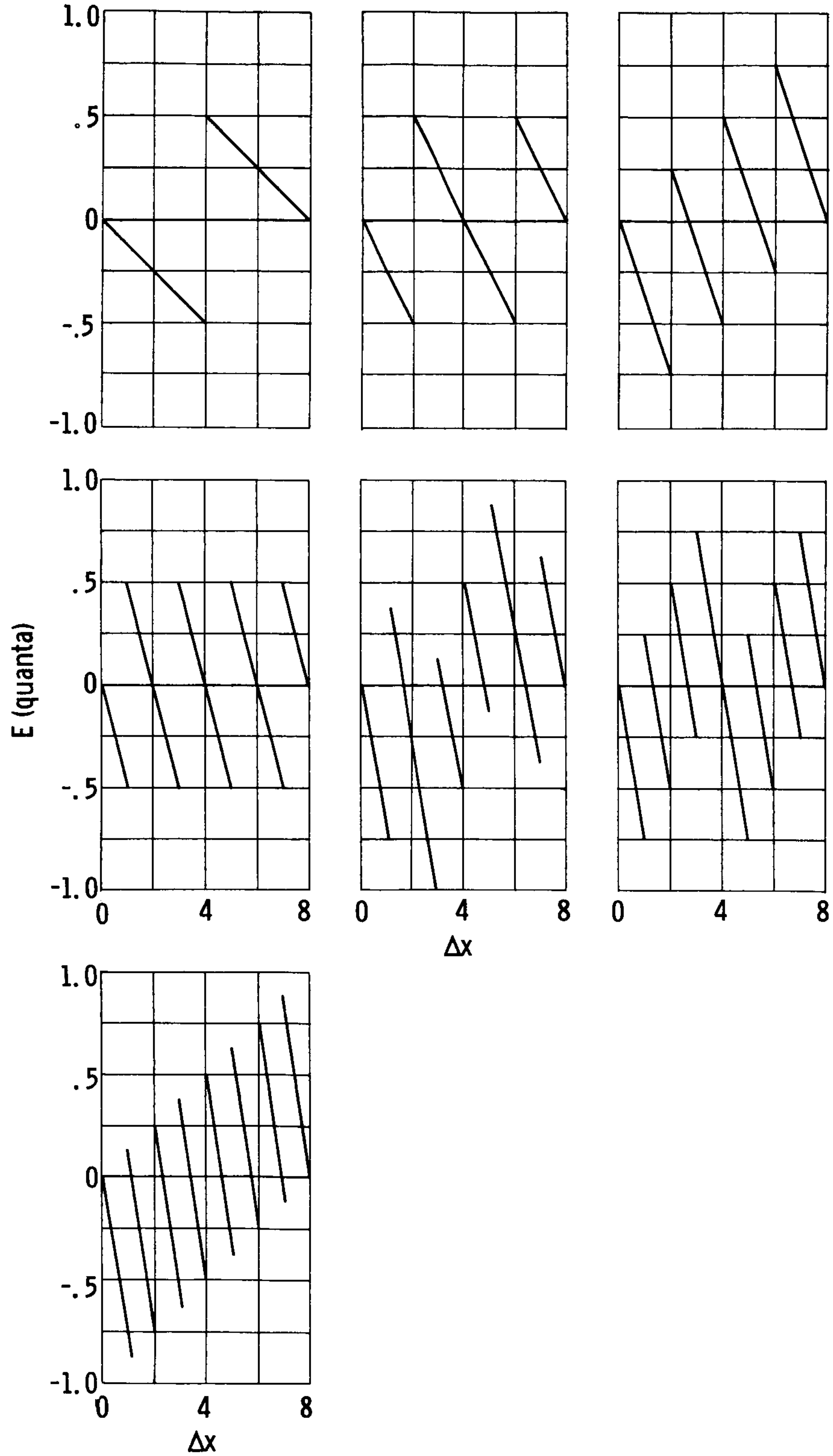


Figure 2.4. - Plot of E for three stage BRM.

TABLE 2.1 - MULTIPLICATION ERROR E OF BRM

WHEN ONE STAGE IS GATED

$y_{-1} = 1$		$y_{-2} = 1$		$y_{-3} = 1$		$y_{-k} = 1$	
x_1	E	x_2x_1	E	$x_3x_2x_1$	E	$x_kx_{k-1} \ldots x_3x_2x_1$	E
0	0	0 0	0	0 0 0	0	0 0 . . . 0 0 0	0
	1/2	0 1	-1/4	0 0 1	-1/8	0 0 . . . 0 0 1	$-1/2^k$
		1 0	2/4	0 1 0	-2/8	0 0 . . . 0 1 0	$-2/2^k$
		1 1	1/4	0 1 1	-3/8	.	.
				1 0 0	4/8	.	.
				1 0 1	3/8	.	.
				1 1 0	2/8	0 1 . . . 1 1 1	$-(2^{k-1}-1)/2^k$
				1 1 1	1/8	1 0 . . . 0 0 0	1/2
						1 0 . . . 0 0 1	$(2^{k-1}-1)/2^k$
						.	.
						.	.
						.	.
						1 1 . . . 1 1 1	$1/2^k$

An inspection of these tables shows that the error associated with the various stages of a BRM may be expressed more concisely in algebraic form as shown in Table 2.2.

TABLE 2.2 - MULTIPLICATION ERROR E

IN ALGEBRAIC FORM

Stage	E
1	$y_{-1}(x_1/2)$
2	$y_{-2}(x_2/2 - x_1/4)$
3	$y_{-3}(x_3/2 - x_2/4 - x_1/8)$
.	.
.	.
.	.
k	$y_{-k}(x_k/2 - x_{k-1}/4 - \ldots - x_1/2^k)$

For an arbitrary value of y, the value of E is the linear combination of the values shown in Table 2.2. This bilinear form is shown in Eq. (2.8) for an n stage BRM. The element subscripts of the Boolean vectors x and y (i.e., vectors whose elements are

0 or 1) which are shown in this equation correspond to the stage numbers of the BRM shown in Fig. 1.1.

$$E = (x_1, x_2, \ . \ . \ . \ , x_n) M \begin{pmatrix} y_{-1} \\ y_{-2} \\ \cdot \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \qquad (2.8)$$

$$M = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} & -\frac{1}{8} & -\frac{1}{16} & \cdots & -\frac{1}{2^{n-1}} & -\frac{1}{2^n} \\ 0 & \frac{1}{2} & -\frac{1}{4} & -\frac{1}{8} & \cdots & -\frac{1}{2^{n-2}} & -\frac{1}{2^{n-1}} \\ \cdot & \cdot & \frac{1}{2} & -\frac{1}{4} & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \frac{1}{2} & \cdots & \cdot & \cdot \\ \cdot & \cdot & & & \ddots & \cdot & \cdot \\ & & & & \frac{1}{2} & -\frac{1}{4} & -\frac{1}{8} \\ & & & & & \frac{1}{2} & -\frac{1}{4} \\ 0 & \cdot & \cdot & \cdot & & 0 & \frac{1}{2} \end{pmatrix} \qquad (2.9)$$

In the formulation of E, the maximum values of the output of the BRM were reflected at the points of discontinuities. It will be observed that just prior to these points the error is one quanta less than that shown by E. A formulation of F in which the minimum values are reflected at the points of discontinuities can be obtained in a manner similar to that for obtaining E. The quantity F, when only one stage of an n stage BRM is gated by

y, is shown in tabular form in Table 2.3 (only a few cases are exhibited).

TABLE 2.3 - MULTIPLICATION ERROR F WHEN

ONLY ONE STAGE IS GATED

$y_{-1} = 1$			$y_{-2} = 1$			$y_{-3} = 1$		
C_1	x_1	N	C_2C_1	x_2x_1	N	$C_3C_2C_1$	$x_3x_2x_1$	N
0	0	0	0 0	0 0	0	0 0 0	0 0 0	0
1	1	-1/2	1 1	0 1	-1/4	1 1 1	0 0 1	-1/8
			1 0	1 0	-2/4	1 1 0	0 1 0	-2/8
			0 1	1 1	1/4	1 0 1	0 1 1	-3/8
						1 0 0	1 0 0	-4/8
						0 1 1	1 0 1	3/8
						0 1 0	1 1 0	2/8
						0 0 1	1 1 1	1/8

It will be observed that the F values equal the E values except at the points where the discontinuity occurs. At these points F equals -1/2 while the corresponding E value equals +1/2. The C values shown above correspond to the 2's complement of the x values. It will be observed that the F values are identical in terms of C to the negative values of E. Consequently, it can be asserted that F in terms of C is just the negative of E.

$$F = -(C_1, C_2, \ldots, C_n) M \begin{pmatrix} y_{-1} \\ y_{-2} \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \tag{2.10}$$

An example will help clarify these formulas. In this example the value of y is 101, and the values of E and F are calculated for successive BRM counter values.

$$E = \begin{pmatrix} 000 \\ 100 \\ 010 \\ 110 \\ 001 \\ 101 \\ 011 \\ 111 \end{pmatrix} \begin{pmatrix} 1/2 & -1/4 & -1/8 \\ 0 & 1/2 & -1/4 \\ 0 & 0 & 1/2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 3/8 \\ -1/4 \\ 1/8 \\ 1/2 \\ 7/8 \\ 1/4 \\ 5/8 \end{pmatrix}$$

$$F = -\begin{pmatrix} 000 \\ 111 \\ 011 \\ 101 \\ 001 \\ 110 \\ 010 \\ 100 \end{pmatrix} \begin{pmatrix} 1/2 & -1/4 & -1/8 \\ 0 & 1/2 & -1/4 \\ 0 & 0 & 1/2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ -5/8 \\ -1/4 \\ -7/8 \\ -1/2 \\ -1/8 \\ 1/4 \\ -3/8 \end{pmatrix}$$

This example is shown in graphical form in Fig. 2.5.

Consider the difference E - F. For a three stage BRM this is:

$$E - F = (x_1, x_2, x_3)M \begin{pmatrix} y_{-1} \\ y_{-2} \\ y_{-3} \end{pmatrix} + (C_1, C_2, C_3)M \begin{pmatrix} y_{-1} \\ y_{-2} \\ y_{-3} \end{pmatrix} \qquad (2.11)$$

The premultiplier to the vector y; i.e.,

$$(x_1, x_2, x_3)M + (C_1, C_2, C_3)M = (x_1 + C_1, x_2 + C_2, x_3 + C_3)M$$

is equal to the values given in the p-sequence.

$$\begin{pmatrix} 000 \\ 211 \\ 021 \\ 211 \\ 002 \\ 211 \\ 021 \\ 211 \end{pmatrix} \begin{pmatrix} 1/2 & -1/4 & -1/8 \\ 0 & 1/2 & 1/4 \\ 0 & 0 & 1/2 \end{pmatrix} = \begin{pmatrix} 000 \\ 100 \\ 010 \\ 100 \\ 001 \\ 100 \\ 010 \\ 100 \end{pmatrix}$$

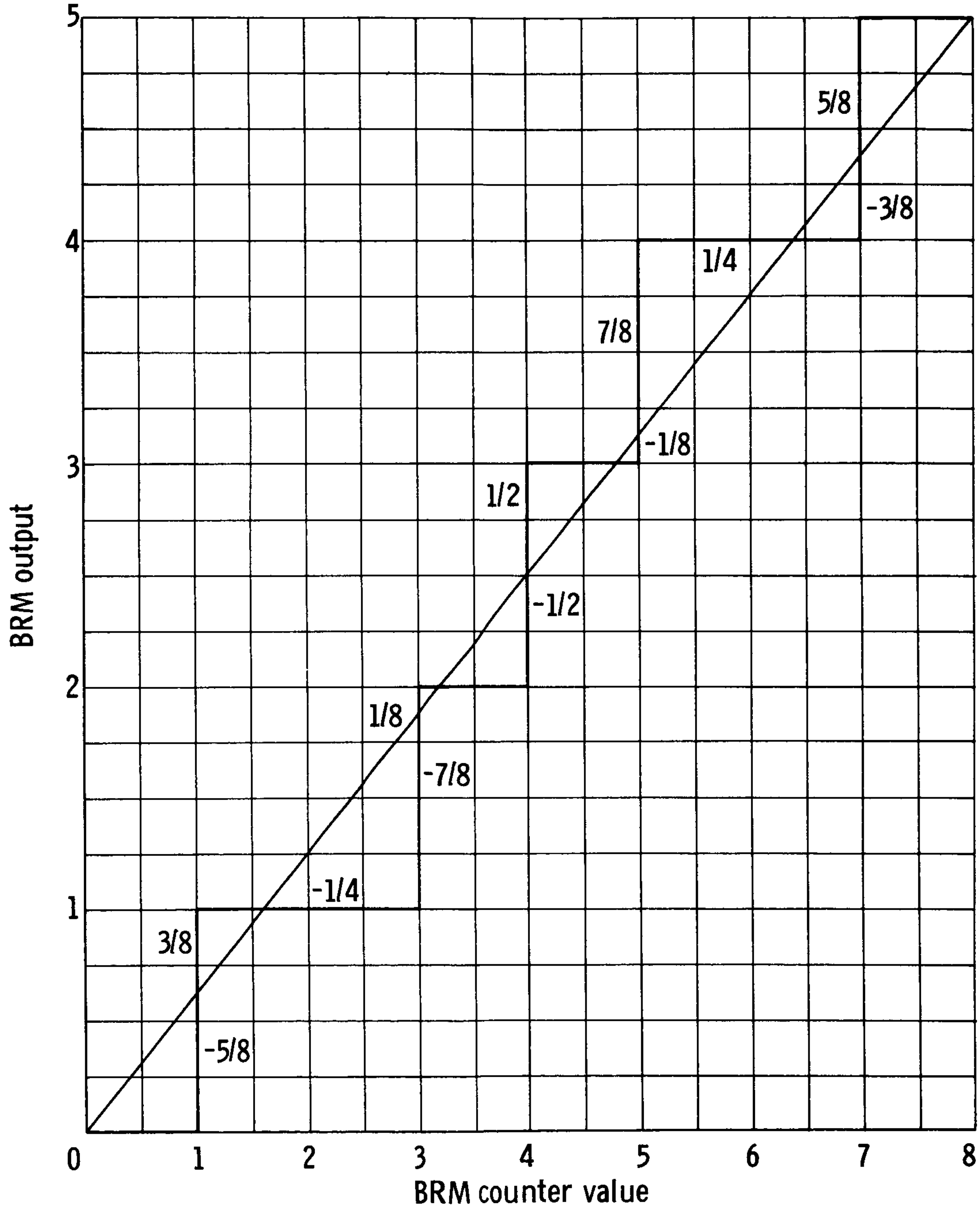


Figure 2.5. - Multiplication error resulting from y = .101.

When a BRM counter starts out with an arbitrary value, then the starting value must enter into the error formula as a parameter. These error formulas are given explicitly by Eqs. (2.12) and (2.13). In particular, Eqs. (2.12) and (2.13) reflect the maximum and minimum values of the actual output at the points of discontinuities, respectively. In these formulations x and x_S represents the value and the initial value of the counter, and C and C_S represents the 2's complements of these values. The subscripts on those literals represent, as before, the stage of the BRM. In Eq. (2.13), x_{SR} identifies the right most counter bit whose value is 1; e.g., for the counter value 100 then $x_{SR}y_{-R} = y_{-3}$, for counter value 011 then $x_{SR}y_{-R} = y_{-1}$, etc.

$$G = (x_1 - x_{S1}, x_2 - x_{S2}, \ldots, x_n - x_{Sn})M \begin{pmatrix} y_{-1} \\ y_{-2} \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \qquad (2.12)$$

$$H = -x_{SR}y_{-R} - (C_1 - C_{S1}, C_2 - C_{S2}, \ldots, C_n - C_{Sn})M \begin{pmatrix} y_{-1} \\ y_{-2} \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \qquad (2.13)$$

Multiplication Error Bounds

The maximum positive error and the minimum negative error for a n stage BRM whose counter starts out with zero may be obtained by an analysis of Eqs. (2.8) and (2.10), respectively. These values will then form a bound of the deviation of the BRM from that of exact multiplication. This analysis is presented in Appendix A. It is shown in that analysis that for an n stage BRM these values are:

$$E_{max}(n) = \frac{7}{18} + \frac{n}{6} + \frac{(-1)^n}{9 \cdot 2^n} \tag{2.14}$$

$$F_{min}(n) = -\frac{7}{18} - \frac{n}{6} - \frac{(-1)^n}{9 \cdot 2^n} \tag{2.15}$$

Eq. (2.14) is plotted together with Eq. (2.15) in Fig. 2.6. As a by-product of developing Eqs. (2.14) and (2.15), it was necessary to find the points where these values occurred. These points are tabulated for various BRM in Tables 2.4 and 2.5.

Appendix B presents an analysis of a BRM whose counter starts out with an arbitrary value. The basis of this analysis is to use Eq. (2.12) to obtain the maximum positive error and to use Eq. (2.13) to obtain the minimum negative error. For a n stage BRM these values are:

$$G_{max}(n) = \frac{1}{9} + \frac{n}{3} - \frac{(-1)^n}{9 \cdot 2^{n-1}} \tag{2.16}$$

$$H_{min}(n) = -\frac{7}{9} - \frac{n}{3} - \frac{(-1)^n}{9 \cdot 2^{n-1}} \tag{2.17}$$

These values form a bound for the generated round-off error. Fortunately, these values are taken on at only two points of the BRM (for $n > 2$), and therefore one can expect better results than would be predicted by these values. These values are plotted in Fig. 2.7 and are also presented together with the points at which they occur in Tables 2.6 and 2.7.

The problem which has been considered in Appendices A and B

TABLE 2.4 - x AND y VALUES FOR E_{max}

n	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	E_{max}
2	11	11	11	11	3/4
3	101	101	111	111	7/8
4	1011	1101	1101	1011	17/16
5	10101	10101	11011	11011	39/32
6	101011	110101	110101	101011	89/64
7	1010101	1010101	1101011	1101011	199/128

TABLE 2.5 - x AND y VALUES FOR F_{min}

n	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	F_{min}
2	01	11	01	11	-3/4
3	011	101	001	111	-7/8
4	0101	1101	0011	1011	-17/16
5	01011	10101	00101	11011	-39/32
6	010101	110101	001011	101011	-89/64
7	0101011	1010101	0010101	1101011	-199/128

TABLE 2.6 - x, x_S, AND y VALUES FOR G_{max}

n	$x_{S7} \cdots x_{S1}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	$x_{S7} \cdots x_{S1}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	G_{max}
2	01	10	01	00	11	11	3/4
3	001	110	011	010	101	101	9/8
4	0101	1010	0101	0010	1101	1011	23/16
5	00101	11010	01011	01010	10101	10101	57/32
6	010101	101010	010101	001010	110101	101011	135/64
7	0010101	1101010	0101011	0101010	1010101	1010101	313/128

TABLE 2.7 - x, x_S, AND y VALUES FOR H_{min}

n	$x_{S7} \cdots x_{S1}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	$x_{S7} \cdots x_{S1}$	$x_7 \cdots x_1$	$y_{-1} \cdots y_{-7}$	H_{min}
2	11	01	11	11	01	11	-3/2
3	101	011	101	111	001	111	-7/4
4	1101	0011	1011	1011	0101	1101	-17/8
5	10101	01011	10101	11011	00101	11011	-39/16
6	110101	001011	101011	101011	010101	110101	-89/32
7	1010101	0101011	1010101	1101011	0010101	1101011	-199/64

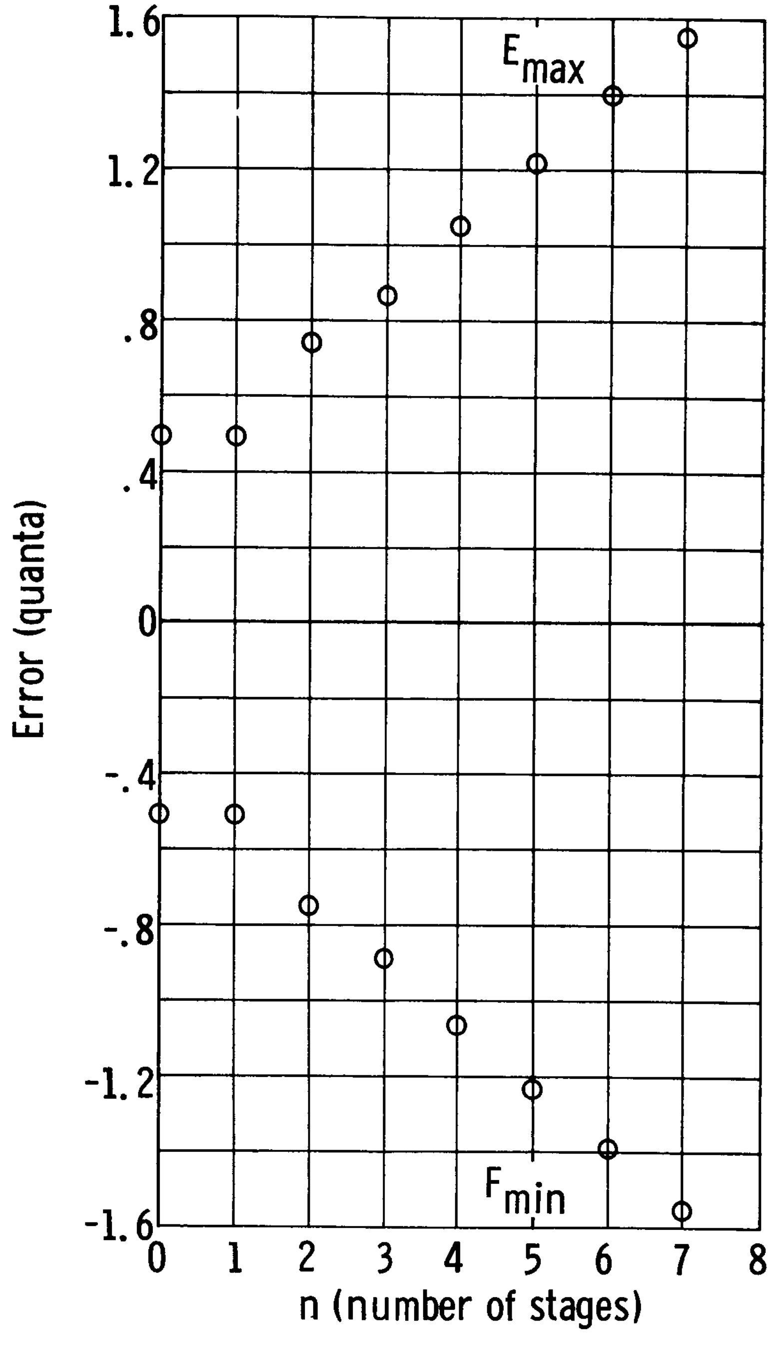


Figure 2.6. - Multiplication error bounds (zero starting).

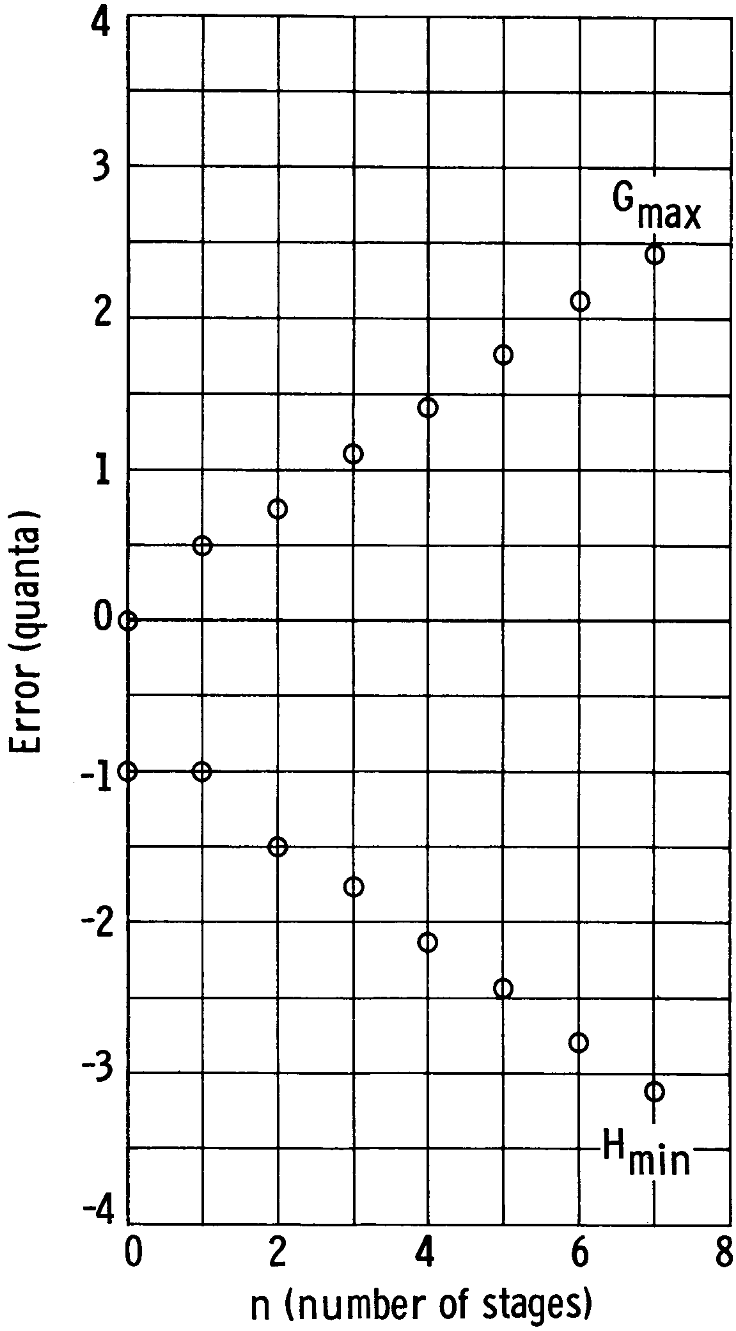


Figure 2.7. - Multiplication error bounds (arbitrary starting).

and in this section is illustrated in Figs. 2.8a and 2.8b. The actual and desired outputs for a 2 and 3 stage BRM are plotted in these figures for all starting values. As is illustrated, finding the multiplication error bounds by graphical means is not trivial. The points labeled E_{max}, F_{min}, G_{max}, and H_{min} in these simple cases agree with those predicted in Appendices A and B.

Error Bounds

The usual formulation of the error problem is to calculate bounds of the total error based on bounds of the generated errors. In the case of Euler's integration these results are available in the literature (e.g., Refs. 15 and 16). Since the analysis by which the bounds are obtained is based on worst case conditions, the results are usually too pessimistic for design purpose. In particular, P. Henrici (Ref. 15) has presented the complete analysis of the initial value problem

$$y' = f(x,y), \quad y(a) = \eta \tag{2.18}$$

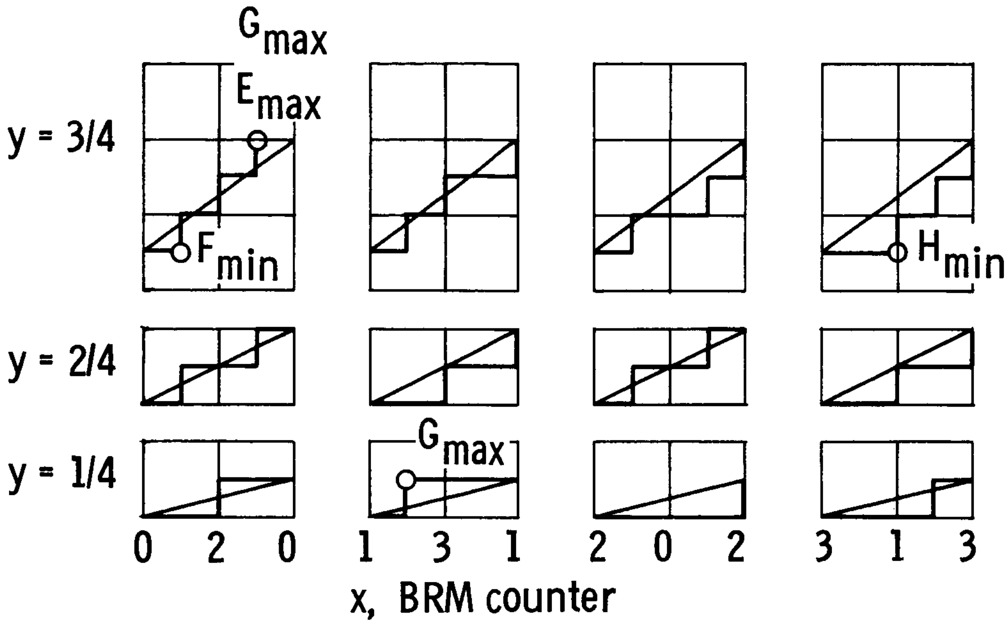
approximated by Euler's integration. The bounds that he presents are in terms of Lipschitz function; i.e.,

$$E_L(x) = \begin{cases} \dfrac{e^{Lx} - 1}{L} & L > 0 \\ x & L = 0 \end{cases} \tag{2.19}$$

where L is a constant such that for any x in the interval $a \leq x \leq b$ and any two values y and y^*

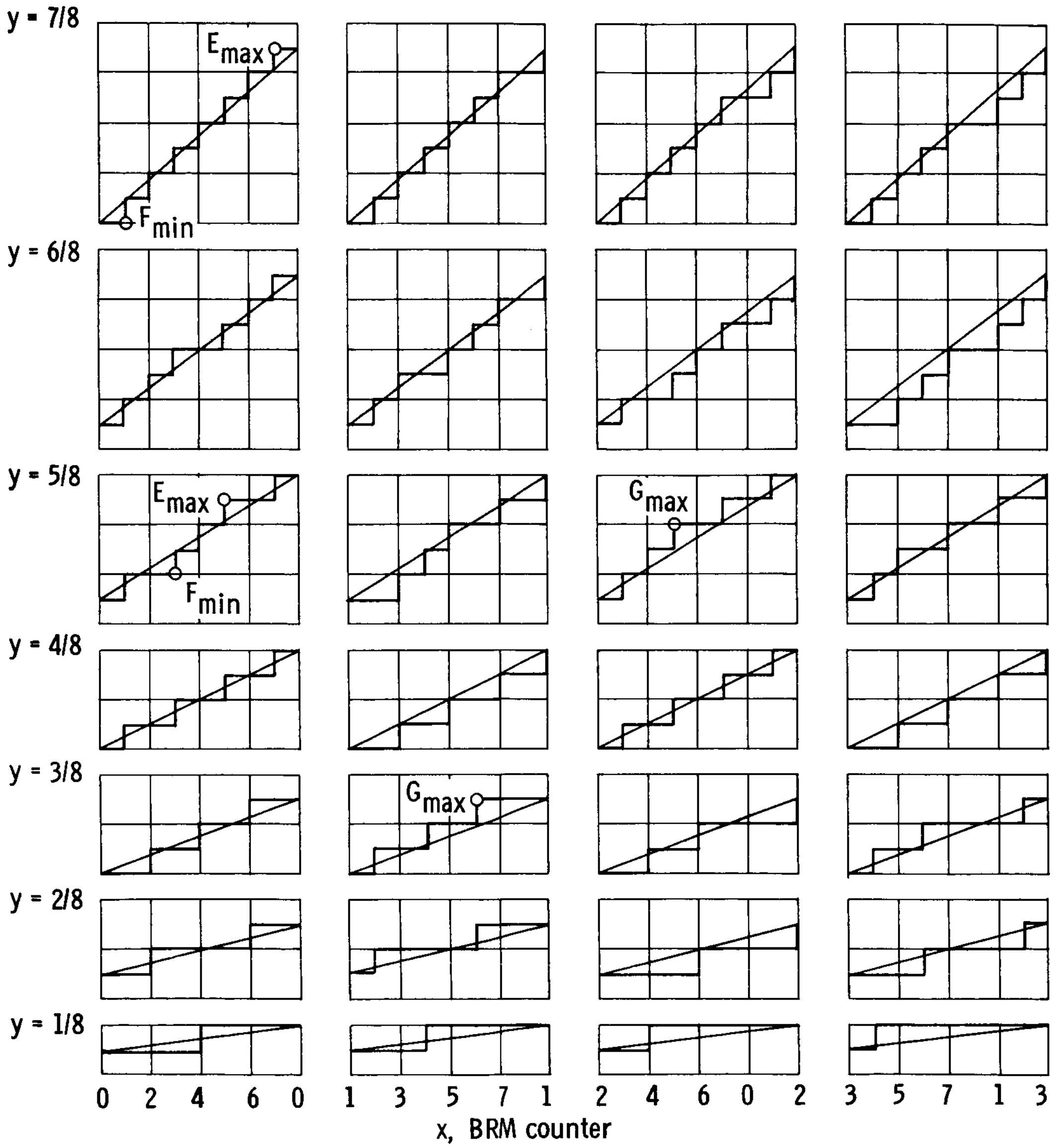
$$|f(x,y) - f(x,y^*)| \leq L|y - y^*| \tag{2.20}$$

The bounds for the truncation error $t_{(k)}$ and the accumulated rounding-off error $r_{(k)}$ are given by



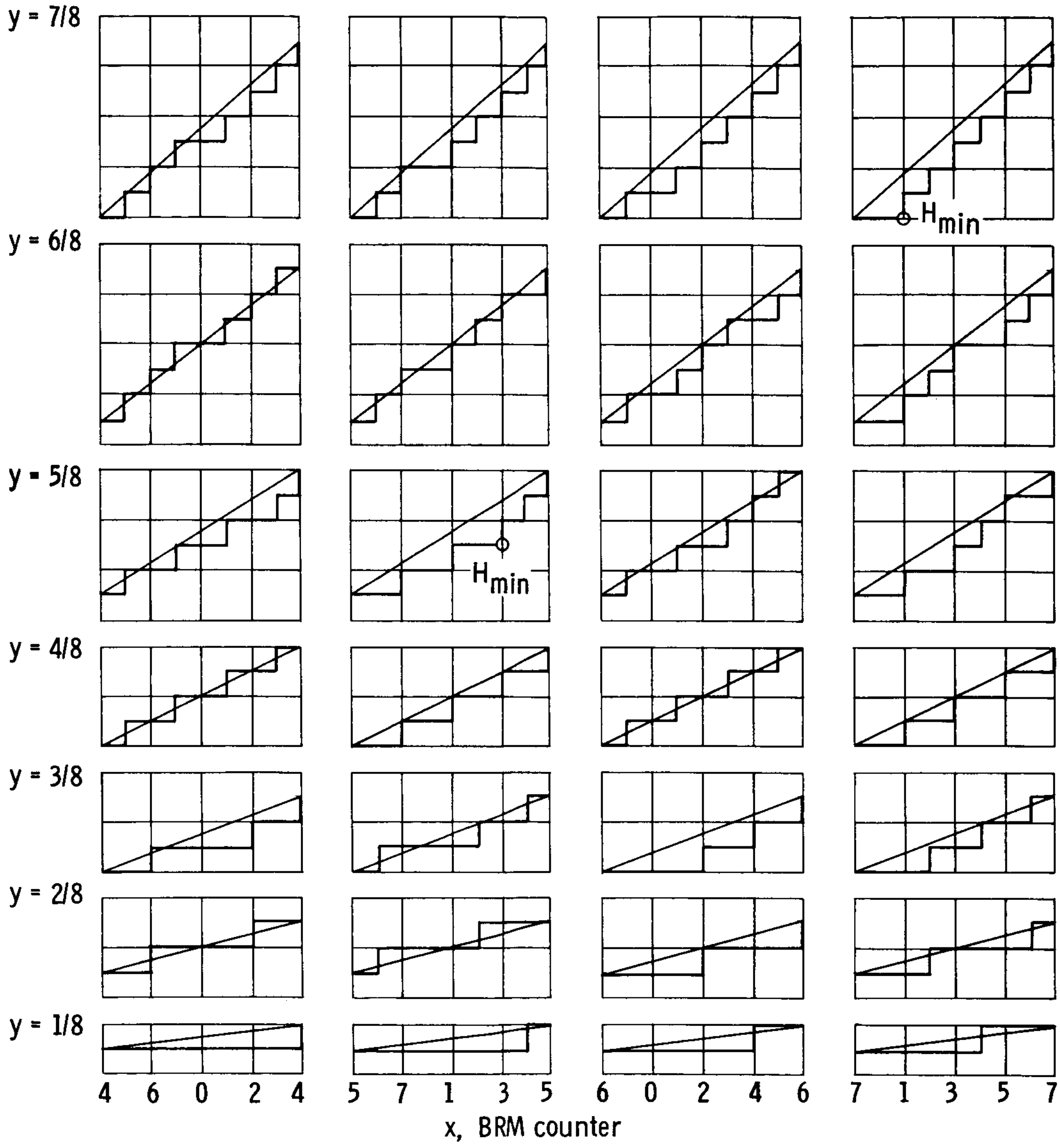
(a) Two-stage.

Figure 2.8. - Output of BRM for all starting values.



(b) Three-stage.

Figure 2.8. - Continued. Output of BRM for all starting values.



(b) Concluded. Three-stage.

Figure 2.8. - Concluded. Output of BRM for all starting values.

$$t_{(k)} \leq \Delta x \; N(x_{(k)}) E_L(x_{(k)} - a) \tag{2.21}$$

$$r_{(k)} \leq \frac{e}{\Delta x} E_L(x_{(k)} - a) \tag{2.22}$$

where $N(x) = 1/2 \max|y''(t)|$ for all t in the interval $a \leq t \leq x$ and e is the maximum local rounding error.

The total error $\mathcal{E}_{(k)}$ is bounded by the sum of these two bounds; i.e.,

$$\mathcal{E}_{(k)} \leq \Delta x \; N(x_k) E_L(x_{(k)} - a) + \frac{e}{\Delta x} E_L(x_{(k)} - a) \tag{2.23}$$

Consider now the generation of this function by use of a BRM. The function $f(x,y)$ is used to set the level of the BRM, and the output of the BRM is summed in a counter which represents the value of y (see Eq. (2.18)). If the value of $f(x,y)$ is updated every Δx pulses, then the bound given by Eq. (2.23) may be applied directly to this process. Suppose the interval size is chosen such that the error bound given by Eq. (2.23) is a minimum. If $f(x,y)$ is updated every pulse instead of every Δx pulses, we could expect that the actual error would be smaller than this minimum. In any event we will take this minimum as the error bound for the function generated by the BRM. The minimum of the right-hand side of Eq. (2.23) is

$$\mathcal{E}_{(k)} \leq \sqrt{e} \; \sqrt{N(x_{(k)})} \; E_L(x_{(k)} - a) \tag{2.24}$$

The value of e in Eq. (2.24) may be obtained from Eq. (2.17). This equation may be used to form a bound of the multiplication error. Moreover for large values of n, this may be approximated by $7/9 + n/3$. If the maximum value of y; i.e., y_{max}, is represented in the counter by 2^n counts, then

$$e \leq \frac{7/9 + n/3}{2^n} y_{max} \tag{2.25}$$

Combining Eqs. (2.24) and (2.25) the error bound is

$$\mathcal{E}_{(k)} \leq \sqrt{\frac{7/9 + n/3}{2^n} y_{max} N(x_{(k)}) E_L(x_{(k)} - a)} \tag{2.26}$$

This equation has the property that the error bound decrease as the number of stages of the BRM is increased. However, it is too pessimistic for design purposes. This point will be illustrated in the next chapter by applying this formula to a specific countup-countdown machine.

CHAPTER III

GENERATION OF FUNCTIONS

Synthesis (Differential Equation)

The method of synthesis which will be applied in this section is to express the function to be generated as the solution to a differential equation. It has been demonstrated with analog techniques that a wide variety of functions can be generated by utilizing only integrating units and adders (e.g., Ref. 8). For example, with a mechanical differential analyser the basic units are a ball-disc integrator and a differential. In the synthesis of countup-countdown machines, the integrator model of Fig. 2.2 and the anti-coincidence circuit which permits the summation of two pulse streams will serve as these operational units. It should be reemphasized that the principle design elements were recognized as entities only for purposes of synthesis, and that the fabrication of the actual machines may permit circuit simplification which may result in reductions of the hardware requirements.

The first step in this synthesis is to express the function to be generated as a differential equation such that the highest order derivative is isolated; i.e.,

$$\frac{d^m y}{dx^m} = f\left(\frac{d^{m-1}y}{dx^{m-1}}, \ . \ . \ ., \ \frac{dy}{dx}, \ y, x\right) \tag{3.1}$$

The independent variable in the above equation is represented by a clock. The next step in the synthesis is to assume that a circuit has been designed to generate the highest order derivative. Integrators may then be used to successively reduce the order of the derivative according to the equation

$$\frac{d^{k-1}y}{dx^{k-1}} = \int \frac{d^k y}{dx^k}\, dx \tag{3.2}$$

A circuit for generating the highest order derivative, whose existence had previously been assumed, may now be developed by the constraint defined by the right hand side of Eq. (3.1). The above process terminates the design of the basic configuration for generating the function.

The schematic diagram of the machine just designed must then be scaled in order to (1) exactly match the defining equation and (2) accommodate the range of variables in a finite machine. In particular, the counters which are used in the machine configuration to handle magnitudes having a finite excursion based on the number of stages. Therefore, when a bidirectional counter is used then its magnitude must be such that

$$|\text{counter value}| \leq 2^n - 1 \tag{3.3}$$

Since the level setting of a BRM must be less than one, then when a counter is used for this purpose its scale will be reduced accordingly, i.e.,

$$[\text{level setting of BRM}] = 2^{-n}\,[\text{counter value}] \tag{3.4}$$

Finally, the scale of both sides of the defining equation; i.e., Eq. (3.1), must be the same.

This procedure may be reduced to a finite number of steps. A synopsis of these steps is as follows:

Step 1. Isolate the highest order derivative in the differential equation as shown in Eq. (3.1).

Step 2. Assume the highest order derivative has been generated in a counter.

Step 3. Generate each successive lower derivative by using an integrating unit.

Step 4. Constraint the independent variable, the function, and its derivatives according to the right hand side of Eq. (3.1) and connect its output directly to the counter representing the highest order derivative.

Step 5. Assign arbitrary constants to the independent variable and its highest order derivative.

Step 6. Write constraint equations at each counter based on the maximum excursion and number of stages.

Step 7. Write constraint equation based on the defining equation.

Step 8. Calculate scale factors to satisfy the equations of Steps 6 and 7.

We continue with the application of this procedure in the design of specific countdown-countup machines. The first two examples will be a machine for generating the exponential function and another machine for generating the sine-cosine functions. These two machines will be illustrated in detail.

Exponential Function

The differential equation

$$y' = y,\ y(0) = 1 \tag{3.5}$$

whose solution is

$$y = e^x \tag{3.6}$$

will be used to design the circuit for generating the exponential function. The design solution for this circuit is presented in Fig. 3.1. The detail procedure for this synthesis is as follows:

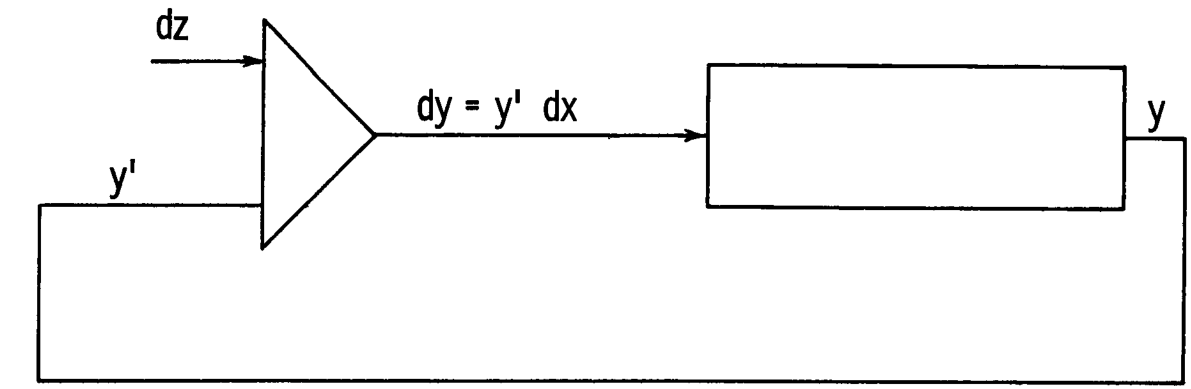
Step 1. The defining differential equation given by Eq. (3.5) is in the desired form.

Step 2. Assume a circuit has been designed to generate the highest order derivative. This is represented by the line labeled y' in Fig. 3.1a.

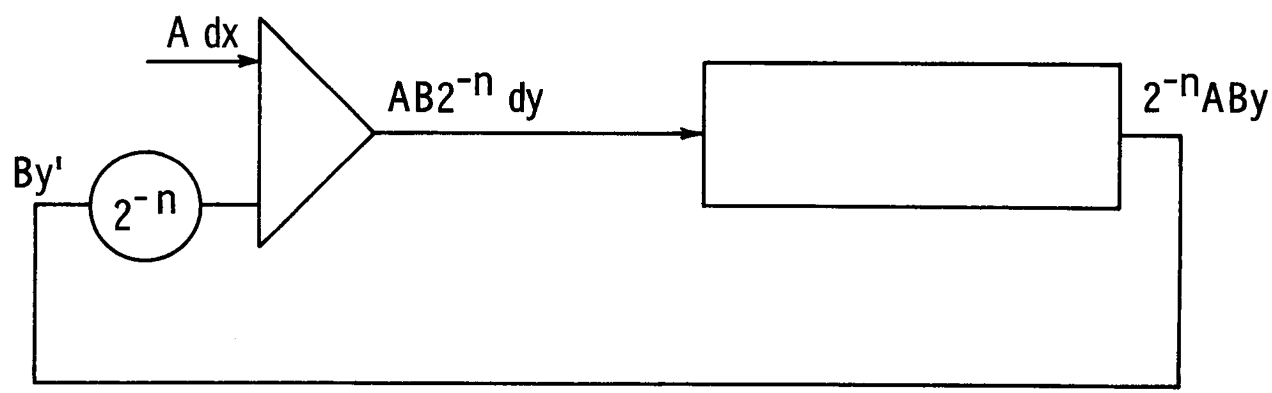
Step 3. The function y is generated by integrating y'.

Step 4. Since by Eq. (3.5) the assumed highest order derivative y' is equal to y, then y is directly connected to the line y'. This completes the basic circuit shown in Fig. 3.1a.

Step 5. This is the first step in the design of the scaled circuit shown in Fig. 3.1b. The arbitrary constants A and B are assigned as scale factors to the independent variable and the highest order derivative, respectively. The interpretation of A is "A counts per unit of x". The interpretation of B is "B counts per unit of y'" Note that the scale factor of the counter in Fig. 3.1b is reduced by 2^{-n} when it is used to set the levels of the BRM. If the value of the counter is By' counts, then the level setting of the BRM is $2^{-n}By'$.



(a) Basic design.



(b) Scaled design.

Figure 3. 1. - Exponential function generator.

Step 6. Constraint equations are written for the counter; i.e.,

$$|2^{-n}ABy|_{max} \leq 2^n - 1 \tag{3.7}$$

Step 7. The mechanization of the defining equation is justified by the following constraint equation.

$$By' = 2^{-n}ABy \tag{3.8}$$

Step 8. From Eq. (3.8) it can be calculated that

$$A = 2^n \text{ counts/unit of } x \tag{3.9}$$

The calculation of B depends on the maximum excursion of the variable y according to the equation

$$B|y_{max}| \leq 2^n - 1 \tag{3.10}$$

This essentially completes the schematic design of the circuit for generating e^x.

In order to select the number of stages; i.e., n, it would be desirable if an equation were available relating n to the accuracy of the machine. Unfortunately, the equation obtained in CHAPTER II is not suitable for this purpose. This may be demonstrated for this machine by calculating the bound given by Eq. (2.26) for various values of n. Using $L = 1$ and $y_{max} = e$ for this process, then

$$E \leq \sqrt{\frac{7/9 + n/3}{2^{n+1}}}\, e(e - 1) = 1.907\sqrt{\frac{7 + 3n}{2^n}} \tag{3.11}$$

Eq. (3.11) may then be used to calculate the bound for this machine. These calculated values are presented in tabular form in Table 3.1 for various values of n.

TABLE 3.1 - ERROR BOUND FOR
EXPONENTIAL MACHINE

n	Bound
6	1.190
7	.892
8	.664
9	.491
10	.362

This bound does have the property, which was observed earlier, of decreasing as n increases. Unfortunately, it does not decrease rapidly enough for design purposes.

Using a value of $y_{max} < 3.2$, some values of B have been calculated and are given in Table 3.2 together with the initial value of the counter to match the initial condition $y(0) = 1$.

TABLE 3.2 - SCALE FACTORS AND INITIAL CONDITIONS
OF EXPONENTIAL MACHINE

n	A	B	Initial counter value
5	32	10	10
6	64	20	20
7	128	40	40
8	256	80	80

This series of machines have been simulated on a computer and the results are presented in Fig. 3.2. When these results are compared to the desired output it is immediately observed that these results are much better than those predicted by the error bounds given by Eq. (3.11). Moreover, the five and seven stage machines are seen to give better results than the six and eight stage machines.

A more realistic evaluation of the results presented in Fig. 3.2 would be to compare them to the difference equation solution. The difference equation for the exponential machine may be

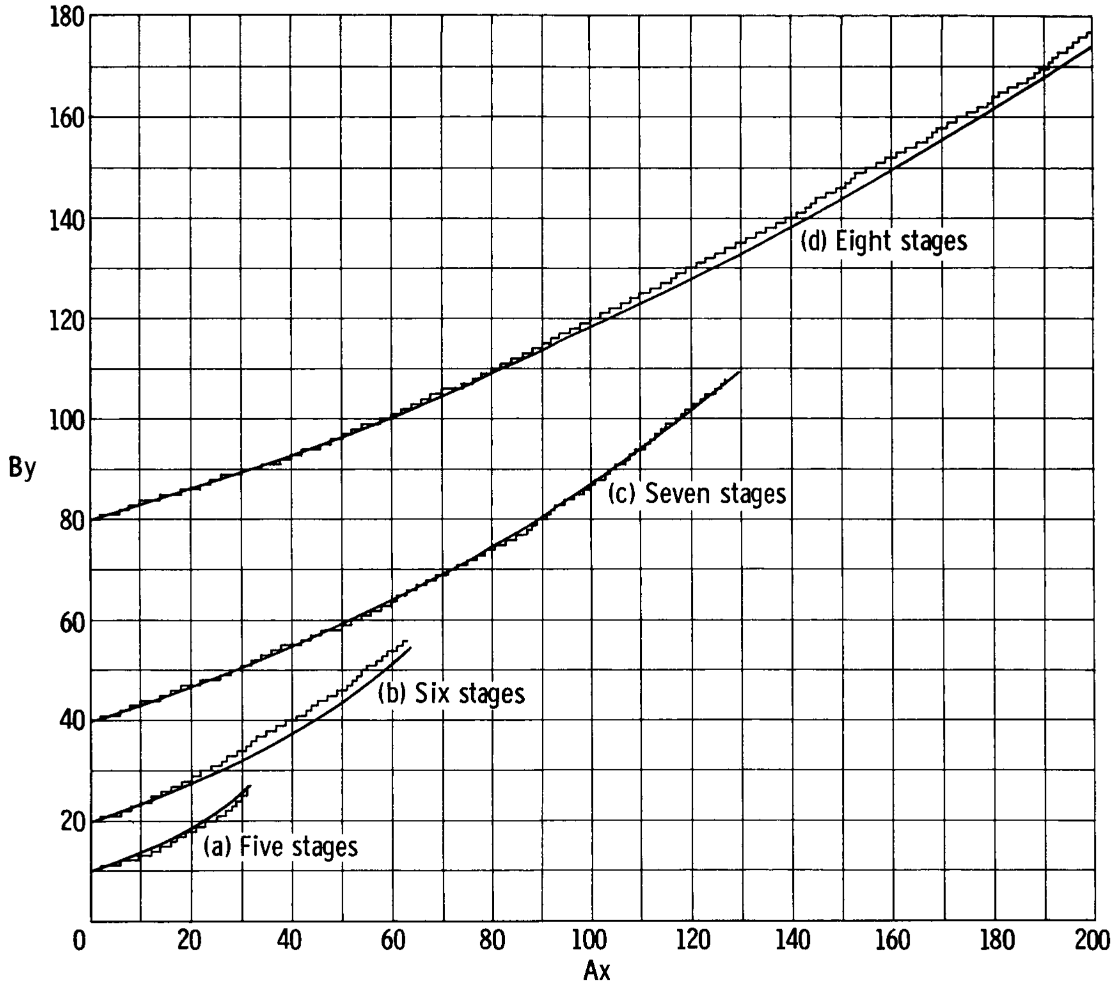


Figure 3.2. - Exponential machine output.

obtained directly from Fig. 3.3, which is identical to the circuit shown in Fig. 3.1 but labeled according to Euler's integration. The scale factor associated with the counter is "B pulses per unit of y." When the counter is at iterative step $k-1$, then its value is at $y_{(k-1)}$. During this iteration $2^n \Delta x$ pulses arrive at the BRM counter and the BRM puts out $By_{(k-1)} \Delta x$ pulses. These output pulses are added to the counter to form the counter value for iterative step k. Mathematically the value of the counter may be expressed by the difference equation

$$By_{(k)} = By_{(k-1)} + By_{(k-1)} \Delta x \tag{3.12}$$

If each clock pulse is taken as an iterative step then $\Delta x = 2^{-n}$ and Eq. (3.12) may be rewritten as

$$y_{(k)} = y_{(k-1)}(1 + 2^{-n}) \tag{3.13}$$

Solving Eq. (3.13) in terms of the initial conditions of y (i.e., $y(0) = 1$), then

$$y_{(k)} = (1 + 2^{-n})^k \tag{3.14}$$

The difference between the difference equation solution and the differential equation solution is the truncation error of the process. The difference between the difference equation solution and the actual output is error due to round-off and is illustrated by the difference in the curves shown in Fig. 3.2. Since the round-off error has been shown to be dependent on the starting value of the BRM, it can be changed by using a different starting value with the objective of obtaining better agreement between the solution and the actual output. Fig. 3.4 presents the actual output

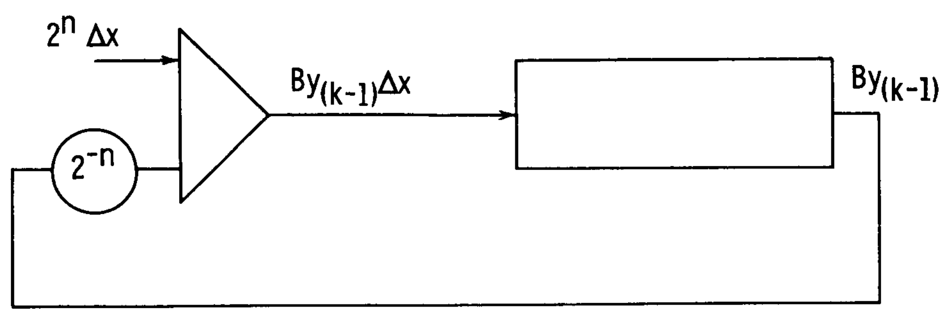


Figure 3.3. - Approximate exponential generator.

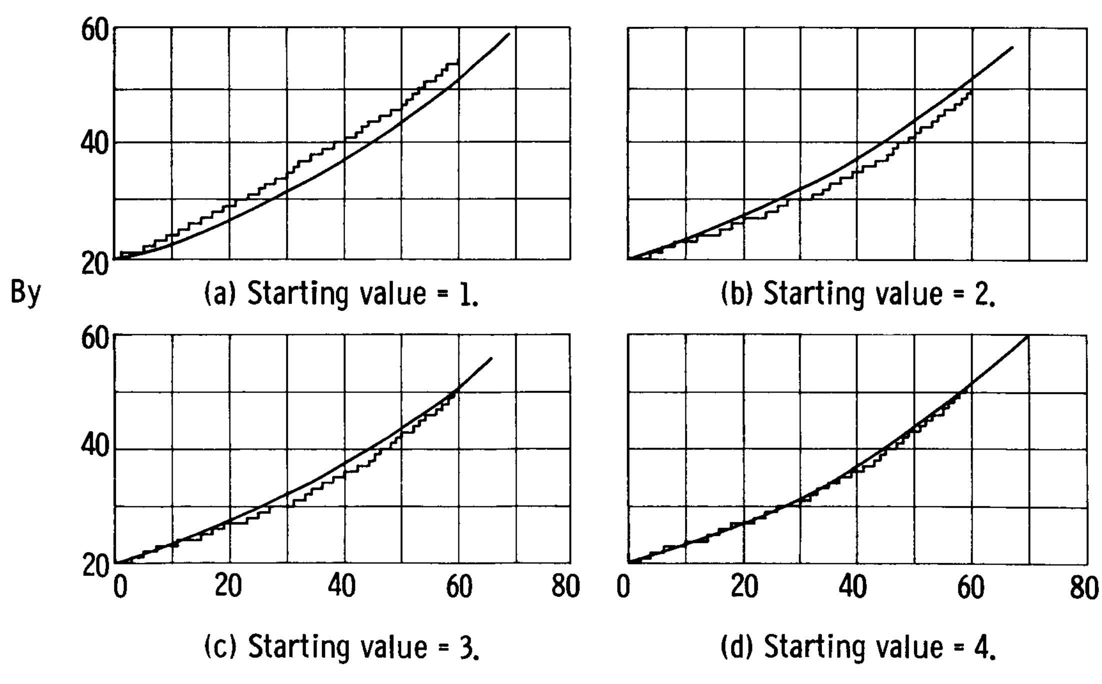


Figure 3.4. - Output of six-stage exponential machine.

for the six stage BRM for a number of starting values together with a plot of the desired results. It is noted that by this simple means measurable improvement has been attained in the total error.

Finally, consider a configuration of this machine built out of logic elements on the Case Logic Breadboard (see Ref. 10) for generating the function. This is shown for a four stage system in Fig. 3.5. Since the exponential function is a monotonic increasing function, the counter shown in this circuit is a simple forward counter.

The synthesis and analysis of a countup-countdown machine for the generation of the general exponential function

$$y = y_0 e^{\alpha x} \tag{3.15}$$

from the differential equation

$$y' = \alpha y, \; y(0) = y_0 \tag{3.16}$$

follows with only minor modification the design of the machine for generating e^x. The schematic diagram for this machine is given in Fig. 3.6a and the logical design is given in Fig. 3.6b.

Sine-Cosine Generator

The differential equation

$$y'' = -y, \; y'(0) = 0, \; y(0) = 1 \tag{3.17}$$

is used to design the sine-cosine generator. The basic schematic diagram and the scaled schematic diagram are shown in Figs. 3.7a and 3.7b, respectively, and may be developed systematically as follows:

<u>Step 1</u>. The defining differential equation given by Eq. (3.17) is in the desired form.

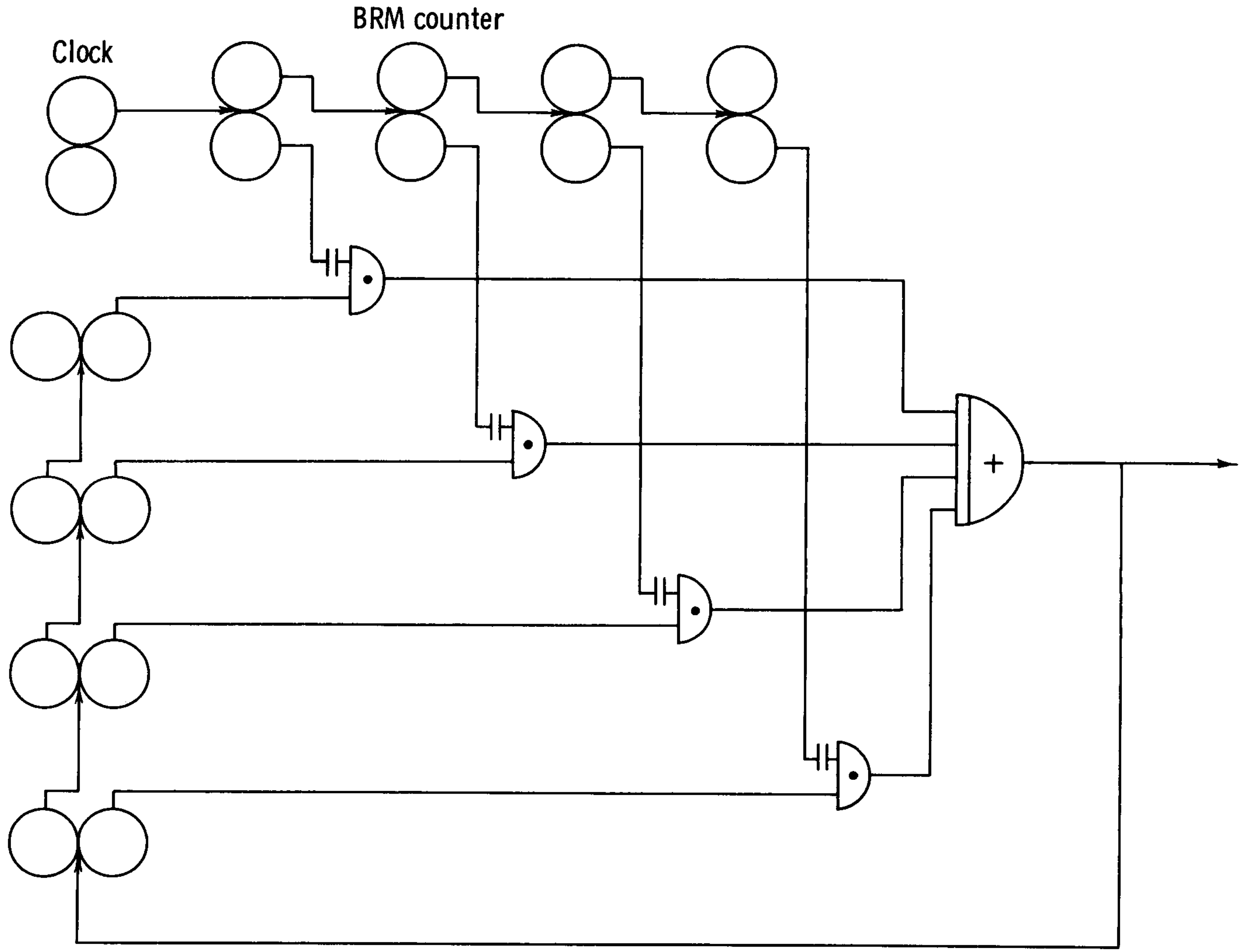
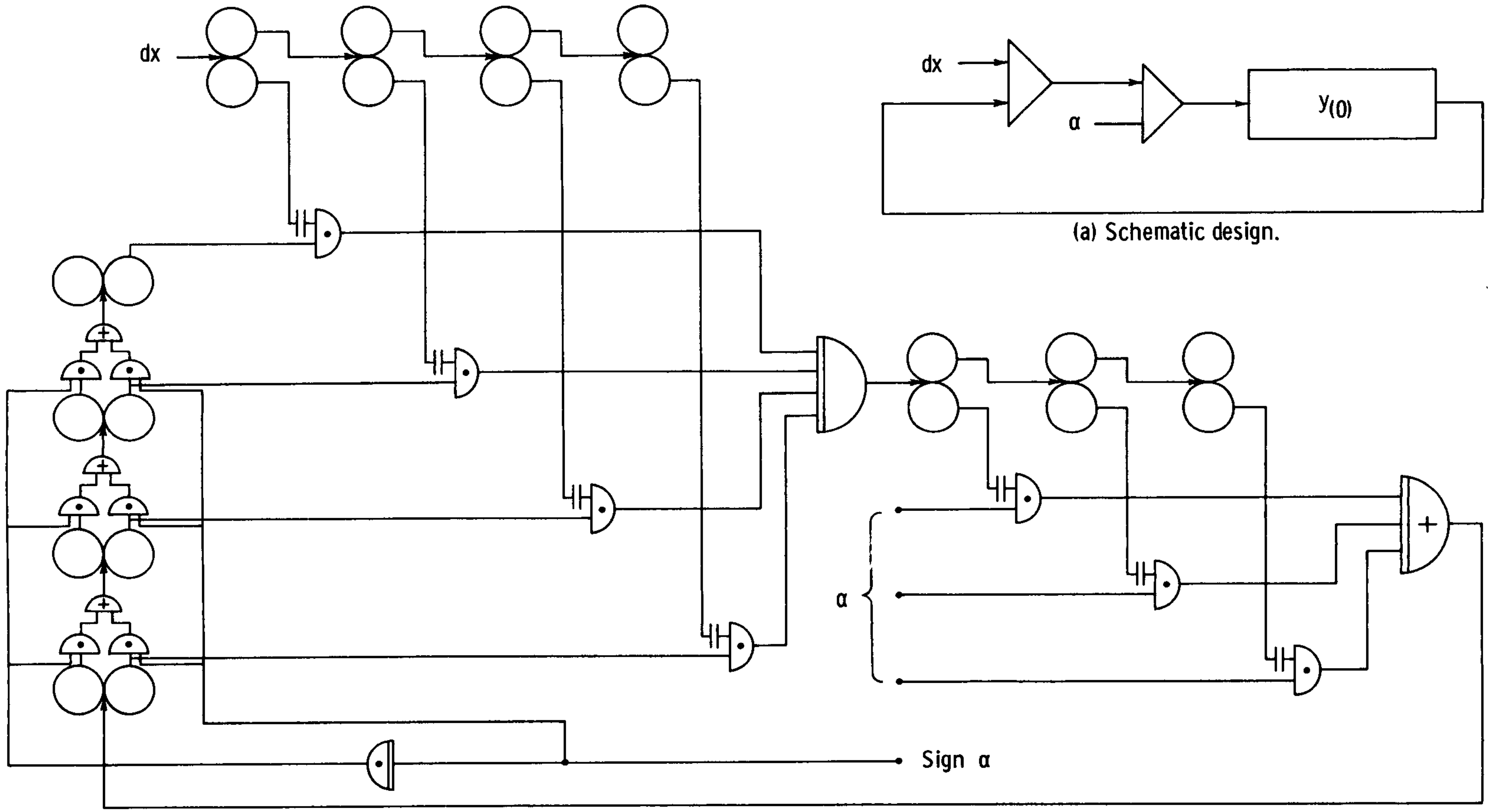


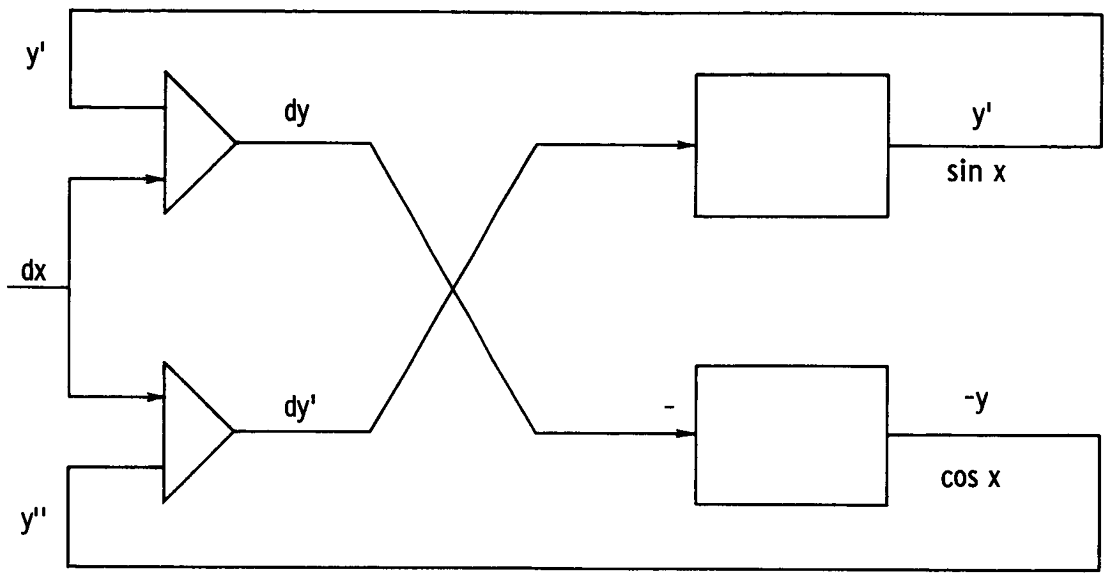
Figure 3.5. - Logic diagram of exponential machine.



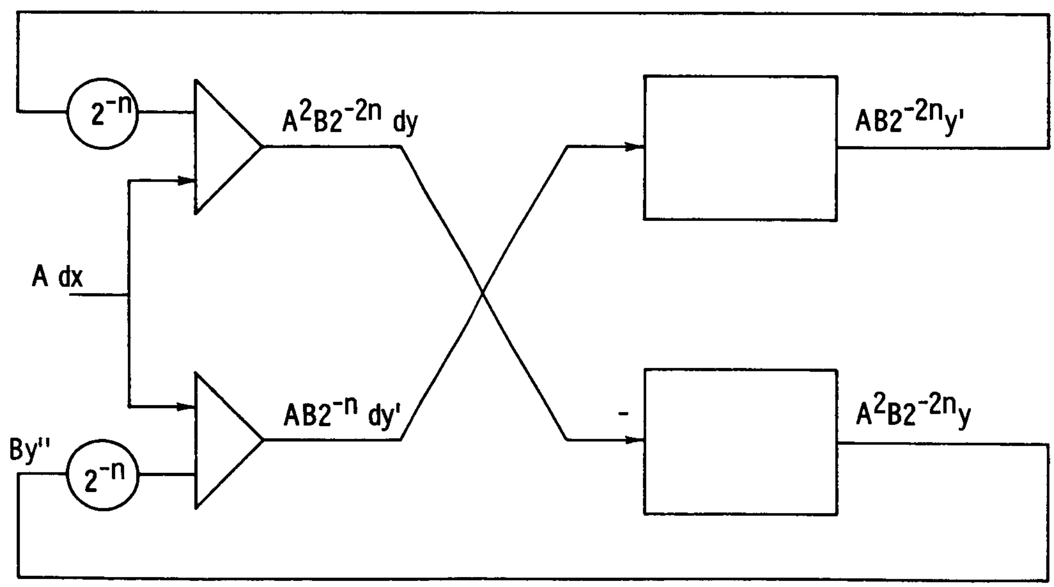
(a) Schematic design.

(b) Logical design.

Figure 3.6. - General exponential function machine.



(a) Basic design.



(b) Scaled design.

Figure 3.7. - Sin-cos generator.

Step 2. Assume that a circuit has been designed to generate the highest order derivative. This is represented by the line labeled y'' in Fig. 3.7a.

Step 3. The function y' is generated by integrating y'', and the function y is generated by integrating y'.

Step 4. Since y enters the differential equation negatively, the pulses arriving at the y counter have a sign change with the result that the output of the counter is $-y$ corresponding to y'' (see defining equation).

Step 5. Arbitrary constants A and B are assigned as scale factors to the independent variable and the highest order derivative, respectively. The interpretation of A is "A counts per unit of x" (i.e., per radian). The interpretation of B is "B counts per unit of y ". If y' counter is set to 0 and y counter is set to -1 (note that this corresponds to the cosine being $+1$), then y' will countup to generate sine and y will countdown to generate the cosine.

Step 6. Constraint equations are written for each counter; i.e.,

$$AB2^{-n}|y'|_{max} \leq 2^n - 1 \tag{3.18}$$

$$A^2B2^{-2n}|y|_{max} \leq 2^n - 1 \tag{3.19}$$

Step 7. Constraint equation is written to justify the defining equation.

$$By'' = -A^2B2^{-2n}y \tag{3.20}$$

Step 8. From Eqs. (3.10) to (3.12), the scale factors can be chosen.

$$A = 2^n, \; B|y|_{max} < 2^n - 1 \tag{3.21}$$

Some choices for A and B are given in Table 3.3 based on Eq. (3.21).

TABLE 3.3 - SCALE FACTORS AND INITIAL CONDITIONS FOR sin-cos MACHINE

n	A	B	sin counter	cos counter
3	8	4	0	-4
4	16	8	0	-8
5	32	16	0	-16
6	64	32	0	-32

This series of machines has been simulated on a computer and the results plotted on Fig. 3.8.

The truncation error associated with this circuit may be calculated by solving the difference equations associated with this circuit. Calling the values of the sine counter and cosine counter at iterative step k "$BS_{(k)}$" and "$BC_{(k)}$," respectively, the difference equations at these two counters are:

$$BS_{(k)} = BS_{(k-1)} + BC_{(k-1)} \Delta x$$

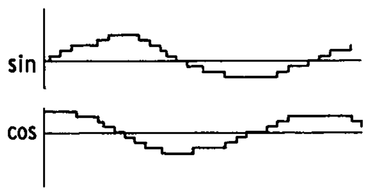
$$BC_{(k)} = BC_{(k-1)} - BS_{(k-1)} \Delta x \tag{3.22}$$

or more concisely

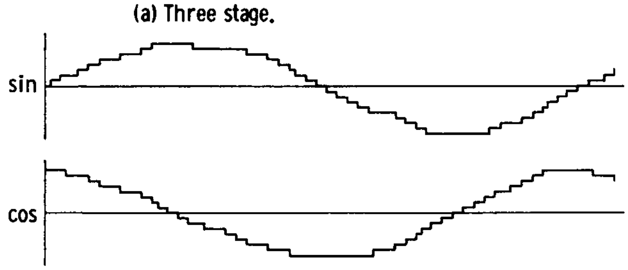
$$\frac{1}{(\Delta x)^2} \nabla^2 S_k = -S_k \tag{3.23}$$

where $\nabla^2 S_k$ is the second backward difference. Therefore, the second derivative in this machine is approximated by the second backward difference.

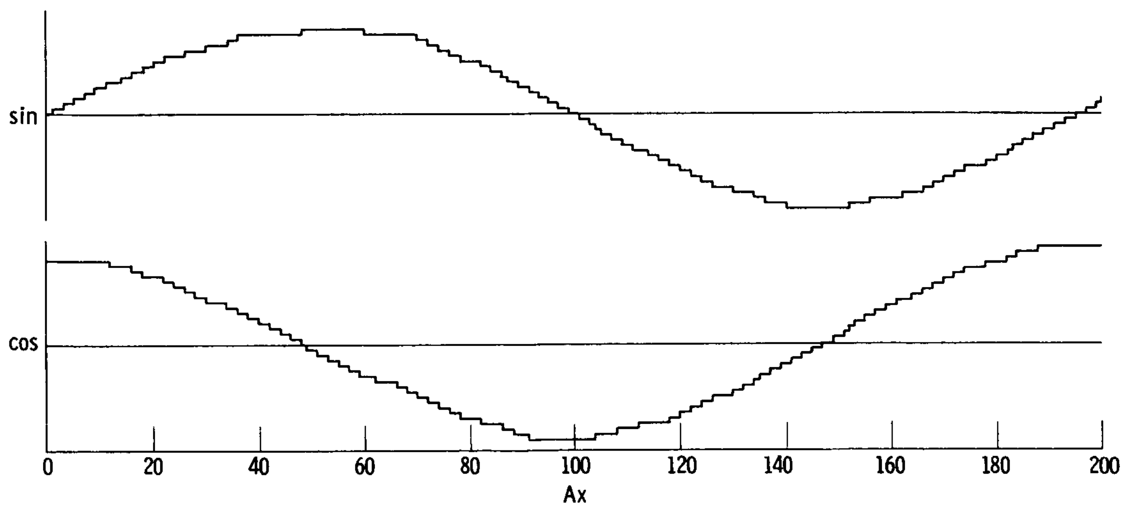
If each clock pulse is taken as in iterative step, then Eq. (3.22) may be written in matrix form as



(a) Three stage.



(b) Four stage.



(c) Five stage.

Figure 3.8. - Output of sin-cos generator.

$$\begin{pmatrix} S_{(k)} \\ C_{(k)} \end{pmatrix} = \begin{pmatrix} 1 & 2^{-n} \\ -2^{-n} & 1 \end{pmatrix} \begin{pmatrix} S_{(k-1)} \\ C_{(k-1)} \end{pmatrix} \tag{3.24}$$

An approximate solution of Eq. (3.24) for large n in terms of the initial conditions

$$\begin{pmatrix} S_{(0)} \\ C_{(0)} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{3.25}$$

may be written as

$$S_{(k)} = (1 + k2^{-2n-1}) \sin 2^{-n}k \tag{3.26}$$

$$C_{(k)} = (1 + k2^{-2n-1}) \cos 2^{-n}k \tag{3.27}$$

A quantitative evaluation of this circuit is complicated by the fact that it is used to generate two functions. One method which seems especially well suited for testing such a circuit is to plot one output function with respect to the other function rather than with respect to the independent variable. For the sin-cos generator this is called the "circle test" since the resultant figure for a perfect sin-cos generator would be a circle. Moreover, it is possible to study the errors due to round-off independent of those due to truncation by comparing the actual output to Eq. (3.24) output. For the sin-cos generator a composite plot of the solution to the difference equation may be simply obtained by expressing Eqs. (3.26) and (3.27) in polar coordinates, with the result that

$$\rho = (1 + 2^{-n-1}\theta) \tag{3.28}$$

The difference between this equation and the circle represents the truncation error of the process and is seen to increase as the spiral of Archemedes. The results plotted in Fig. 3.8 are compared to the plot of Eq. (3.28) in Fig. (3.9) by this method.

Other Differential Equation Machines

The sinh and cosh machine is based on the differential equation

$$y'' = +y \tag{3.29}$$

The schematic diagram for this machine is similar to the sine-cosine generator except that all outputs from the BRM's are added to the counters. The basic circuit and the scaled circuit for this machine is shown in Fig. 3.10. It will be noted from this diagram that the values of A and B are defined by the equation set (3.30) below.

$$\begin{aligned} By'' &= A^2 B 2^{-2n} y \\ A &= 2^n \\ B|y_{max}| &\leq 2^n - 1 \\ B|y'_{max}| &\leq 2^n - 1 \end{aligned} \tag{3.30}$$

The output for a sinh and cosh generator is plotted in Fig. 3.11 for a five stage system where B is chosen equal to 16. It is instructive to display the p-sequence calculation from which these results were obtained. These are shown in Fig. 3.12.

A series of other useful machines will be illustrated in this section. In particular, if two pulse streams du and dv are given, then the product, uv, may be generated by using the equation

$$duv = u\,dv + v\,du \tag{3.31}$$

The basic design for this product machine is shown in Fig. 3.13.

The machine for generating the square of a function is shown schematically in Fig. 3.14. This machine will generate the function

$$y = x^2 \tag{3.32}$$

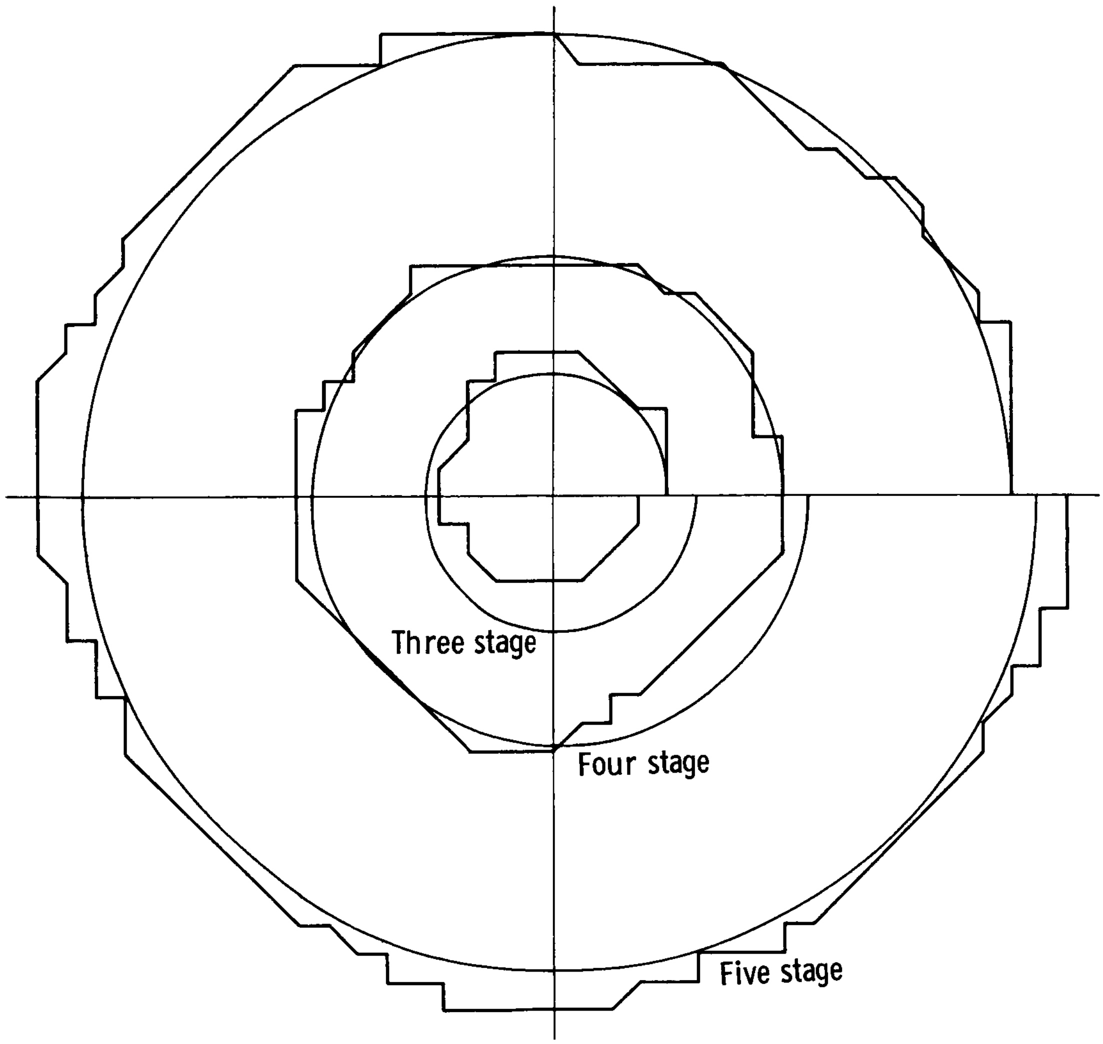
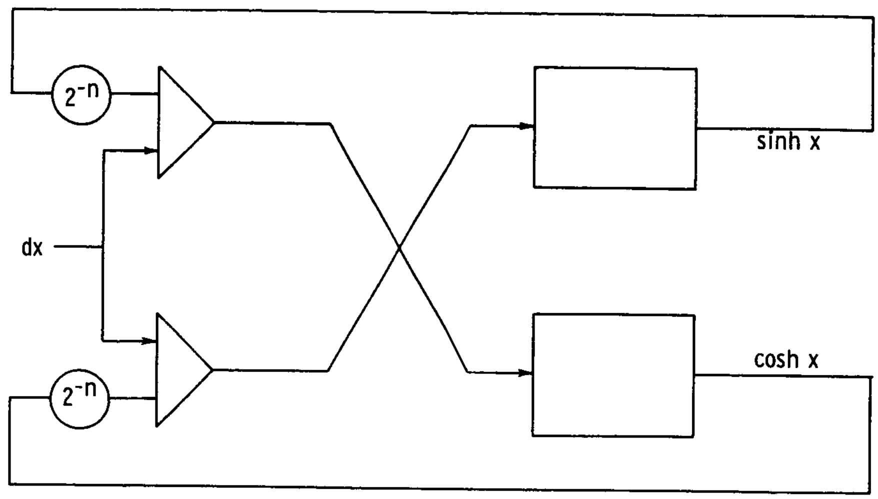
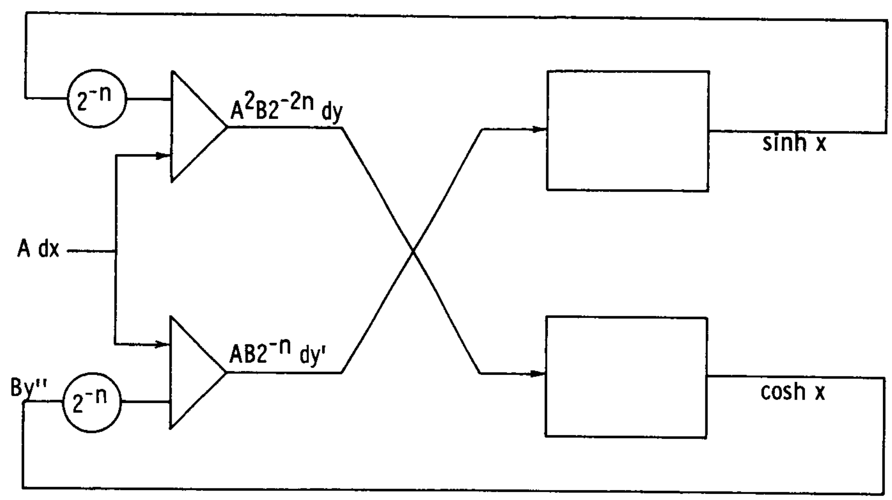


Figure 3.9. - Sin-cos circle test.



(a) Basic design.



(b) Scaled design.

Figure 3.10. - Sinh-cosh generator.

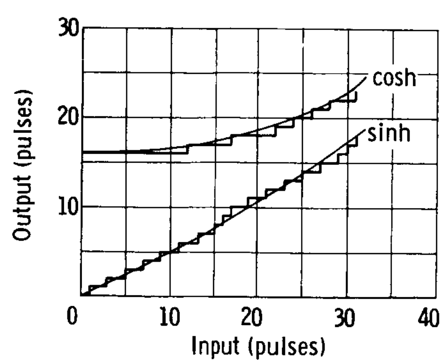


Figure 3.11. - Output of 5 stage sinh-cosh machine.

```
P-sequence  Sinh x
1 0 0 0 0
0 1 0 0 0   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1
1 0 0 0 0   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
0 0 1 0 0   0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0
1 0 0 0 0   0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 1
0 1 0 0 0   0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1
1 0 0 0 0   Output of sinh
0 0 0 1 0   0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0
1 0 0 0 0
0 1 0 0 0
1 0 0 0 0
0 0 1 0 0
1 0 0 0 0   Cosh x
0 1 0 0 0
1 0 0 0 0   1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 0 0 1   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
1 0 0 0 0   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0
0 1 0 0 0   0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 0
1 0 0 0 0   0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0 0 1 1 0 0 1 1 0 0 0 1 1 0
0 0 1 0 0   Output of cosh
1 0 0 0 0   1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 0 1 0 1 1 1 0 1 0
0 1 0 0 0
1 0 0 0 0
0 0 0 1 0
1 0 0 0 0
0 1 0 0 0
1 0 0 0 0
0 0 1 0 0
1 0 0 0 0
0 1 0 0 0
1 0 0 0 0
0 0 0 0 0
```

Figure 3.12. - P-sequence calculations for sinh-cosh machine.

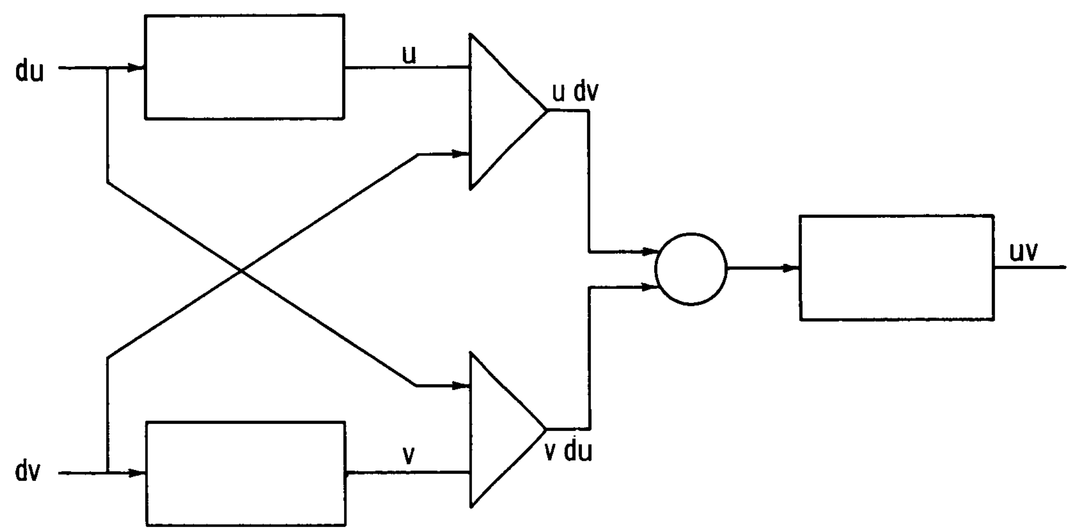


Figure 3.13. - Product machine.

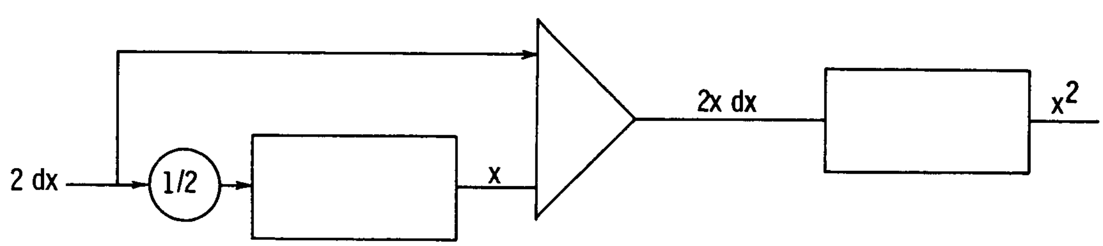


Figure 3.14. - Square machine.

and is based on the equation

$$y' = 2x \tag{3.33}$$

The square machine is utilized as a subassembly in the machine for generating the reciprocal of a function; i.e.,

$$y = 1/x \tag{3.34}$$

The basic design for this reciprocal machine based on the differential equation

$$y' = -y^2 \tag{3.35}$$

is shown in Fig. 3.15.

The machine for generating the solution to the second order differential

$$y'' + 2\omega_n\xi y' + \omega_n^2 y = 0 \tag{3.36}$$

is shown in Fig. 3.16.

The tan machine is shown schematically in Fig. 3.17. This machine is based on the differential equation

$$y' = 1 + y^2 \tag{3.37}$$

A similarity will be noted between this machine and that of the reciprocal machine.

The square root machine is based on the solution of the differential equation

$$y' = -1/(2y) \tag{3.38}$$

It will be noted by this equation that it will form a subassembly of the reciprocal machine. That is, the differential equation

$$dz/dy = 2z^2 \tag{3.39}$$

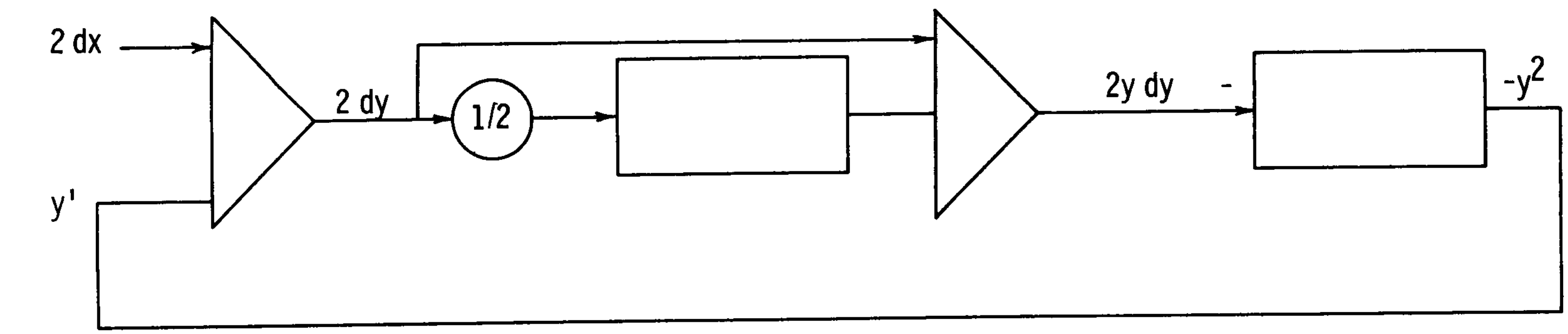


Figure 3.15. - Reciprocal machine.

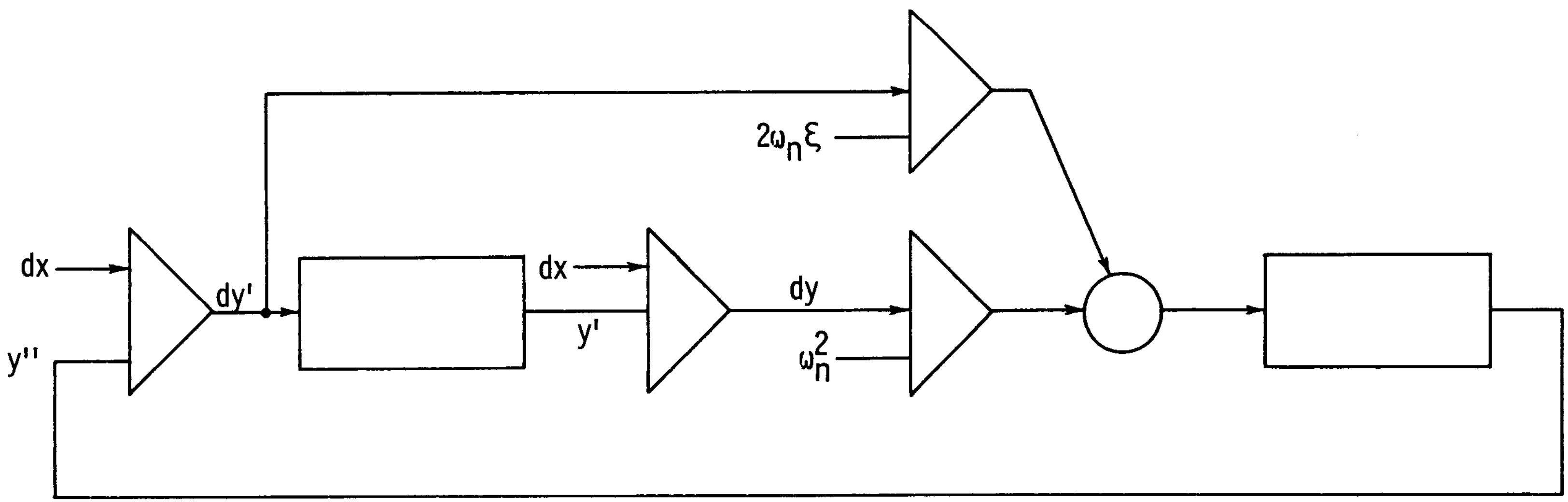


Figure 3.16. - Second order differential equation machine.

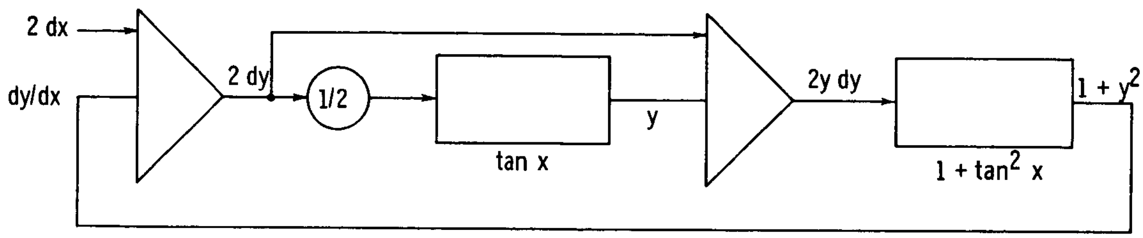


Figure 3.17. - Tan machine.

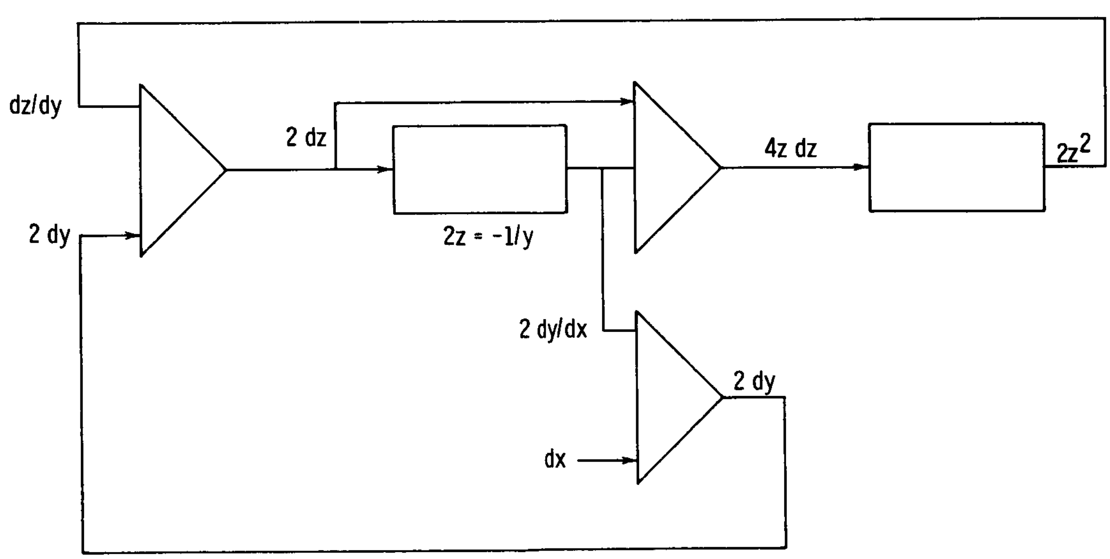


Figure 3.18. - Square root machine.

is used in order to form the function

$$z = -1/(2y) \tag{3.40}$$

The basic design of this machine is shown in Fig. 3.18.

Synthesis (Difference Equation)

Consider the iterative process of successive substitution in the functional equation

$$x_{(k+1)} = \varphi(x_{(k)}) \tag{3.41}$$

where $\varphi(x)$ is chosen such that the fixed points of $\varphi(x)$ (i.e., the points x_i where $x_i = \varphi(x_i)$) are the roots of $f(x) = 0$. One simple form of φ might be $x - f(x)$ which leads to the iterative process

$$x_{(k+1)} = x_{(k)} - f(x_{(k)}) \tag{3.42}$$

A more general form is $x - g(x)f(x)$ which leads to the iterative process

$$x_{(k+1)} = x_{(k)} - g(x_{(k)})f(x_{(k)}) \tag{3.43}$$

A restriction on $g(x)$ in this latter form is that it has no zeros that are not zeros of $f(x)$ and that the multiplicity of its poles at the zeros of $f(x)$ be less than the multiplicity of the zeros of $f(x)$ at these points. With these restrictions, it can be readily seen that the iterative Eq. (3.43) has fixed points at the zeros of $f(x)$ (i.e., $x = x - 0$). The function $g(x)$ in Eq. (3.43) is chosen so that the process converges.

The basic equation of a counter immediately suggests a method for generating an equation of the form of Eq. (3.43). This method of synthesis is simply to generate a pulse stream equal to $g(x)f(x)$

and feed it into a counter. This procedure may be outlined as follows:

Step 1. Write the defining equation in the implicit form $g(x)f(x) = 0$.

Step 2. Assume a counter with the value of x and generate a pulse stream equal to $g(x)f(x)p$, where $g(x)f(x)$ is the level setting of a BRM and p is the input to the BRM counter.

Step 3. Feed back the pulse stream generated in Step 2 into the x counter.

Step 4. Assign an arbitrary constant to each variable represented in the machine.

Step 5. Write constraint equations and calculate the scale factors such that these equations are satisfied.

Divide Algorithm

The machine for generating x such that

$$x = a/b \tag{3.44}$$

may be designed as follows:

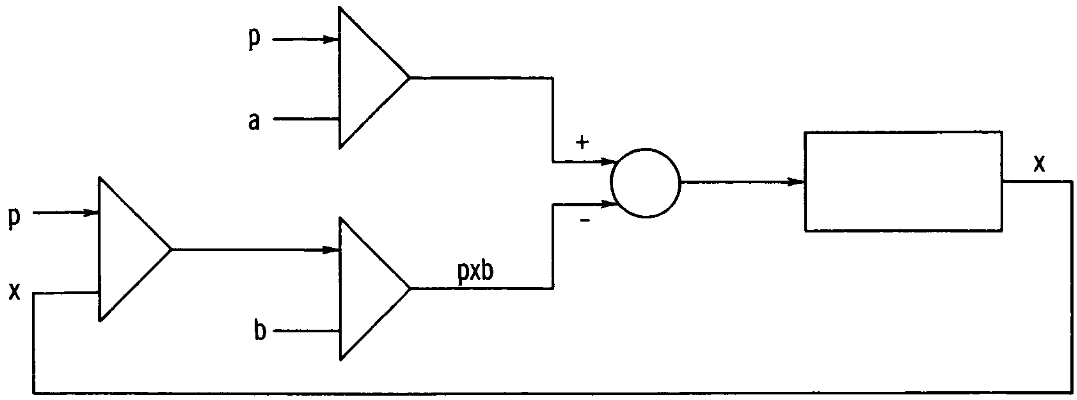
Step 1. One way in which Eq. (3.44) may be rewritten to put it into the desired form is

$$xb - a = 0 \tag{3.45}$$

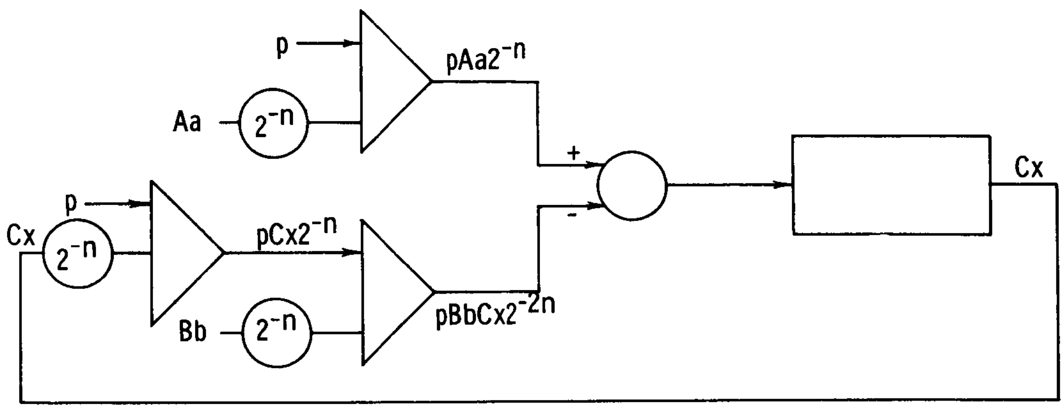
Step 2. Assuming a counter value representing x, a pulse stream equal to $xb - a$ may be generated. This is shown in Fig. 3.19a.

Step 3. The pulse stream generated in Step 2 is fed into the counter representing x.

Step 4. The schematic of Fig. 3.19a is redrawn in Fig. 3.19b, and each variable is assigned an arbitrary constant.



(a) Basic design.



(b) Scaled design.

Figure 3.19. - Divide algorithm.

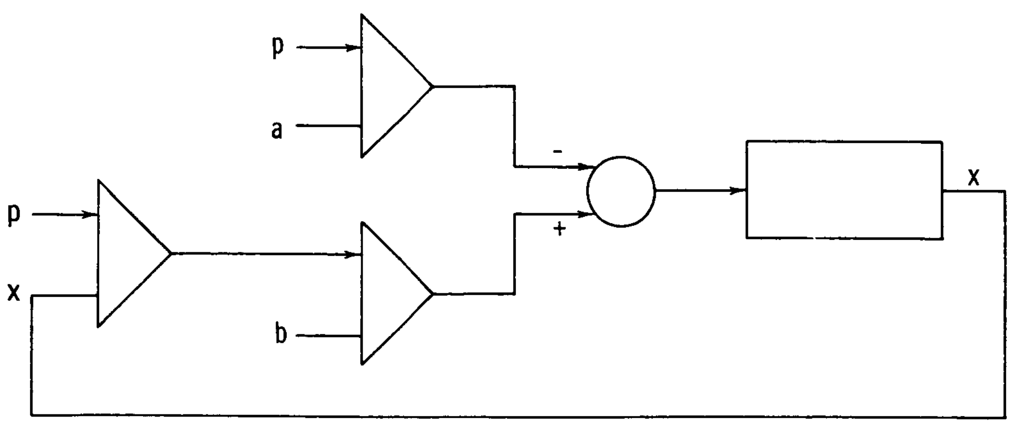


Figure 3.20. - Non-convergent divide algorithm.

Step 5. The constraint equations may be written directly from Fig. 3.19b, in which an iteration is taken every clock pulse.

$$C|x_{max}| \leq 2^n - 1$$

$$A|a_{max}| \leq 2^n - 1$$

$$B|b_{max}| \leq 2^n - 1 \tag{3.46}$$

$$Cx_{(k+1)} = Cx_{(k)} - BbCx_{(k)}2^{-2n} + Aa2^{-n} \tag{3.47}$$

Suppose for the sake of argument that

$$A = B = C = 2^n \tag{3.48}$$

Eq. (3.48) implies that

$$x_{max} \leq 1 - 2^{-n}$$

$$a_{max} \leq 1 - 2^{-n}$$

$$b_{max} \leq 1 - 2^{-n} \tag{3.49}$$

and Eq. (3.47) may be rewritten as

$$x_{(k+1)} = x_{(k)} - 2^{-n}bx_{(k)} + a2^{-n} \tag{3.50}$$

If Eq. (3.50) converges to a fixed point, x, then

$$x = x - 2^{-n}bx + a2^{-n} \tag{3.51}$$

If $\mathcal{E}_{(k)}$ is the iterative truncation error at iterative step k; i.e,

$$\mathcal{E}_{(k)} = x - x_{(k)} \tag{3.52}$$

then from Eqs. (3.50) and (3.51)

$$\mathcal{E}_{(k+1)} = (1 - b2^{-n})\mathcal{E}_{(k)} \tag{3.53}$$

This may be written in terms of $\mathcal{E}_{(0)}$ as

$$\mathcal{E}_{(k)} = (1 - b2^{-n})^k\mathcal{E}_{(0)} \tag{3.54}$$

This process will converge if

$$\lim_{k\to\infty} \mathcal{E}_{(k)} = 0 \tag{3.55}$$

which implies the condition

$$|1 - b2^{-n}| < 1 \tag{3.56}$$

for convergence. Therefore, by Eq. (3.56) the process is seen to converge. However, if the sign of the outputs of the BRM are reversed such as shown in Fig. 3.20, then an analysis will show that the process will not converge.

Other Difference Equation Machines

The square root machine; i.e.,

$$x = \sqrt{a} \tag{3.57}$$

may be designed by finding the zeros of the equation

$$x^2 - a = 0 \tag{3.58}$$

This machine is shown schematically in Fig. 3.21. If the scale factor of x and a are both taken as 2^n, then an analysis similar to that of the divide algorithm shows the iterative process

$$x_{(k+1)} = x_{(k)} - 2^{-n}x^2_{(k)} + 2^{-n}a \tag{3.59}$$

is generated by the machine. The iteration truncation error for this equation may be written as

$$\mathcal{E}_{(k+1)} = 1 - 2^{-n}(x + x_{(k)})\ \mathcal{E}_{(k)} \tag{3.60}$$

where x is the solution. A sufficient condition for this process to converge is that

$$|1 - 2^{-n}(x + x_{(k)})| < 1 \tag{3.61}$$

Since

$$x_{max} < 1 \tag{3.62}$$

with the scale factor chosen, then the process converges, however, had we chosen

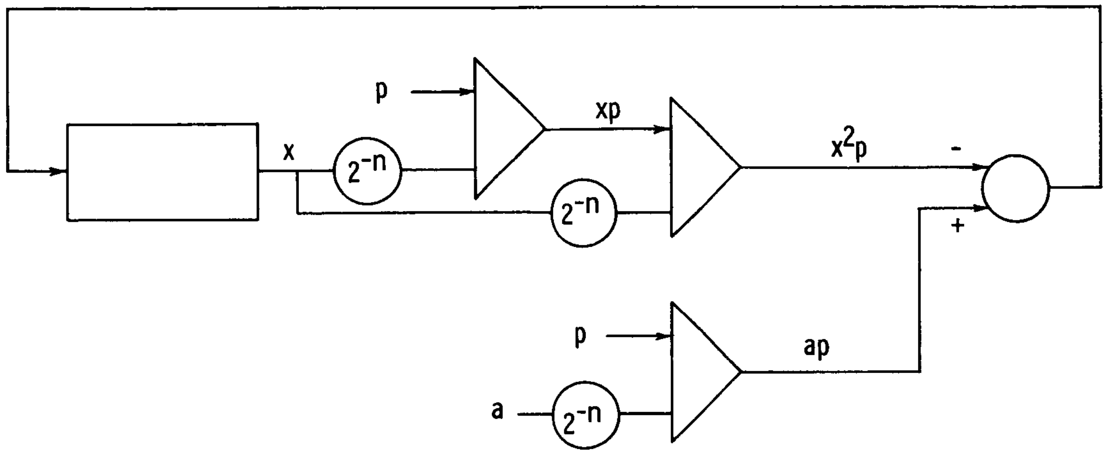


Figure 3.21. - Iterative process square root machine.

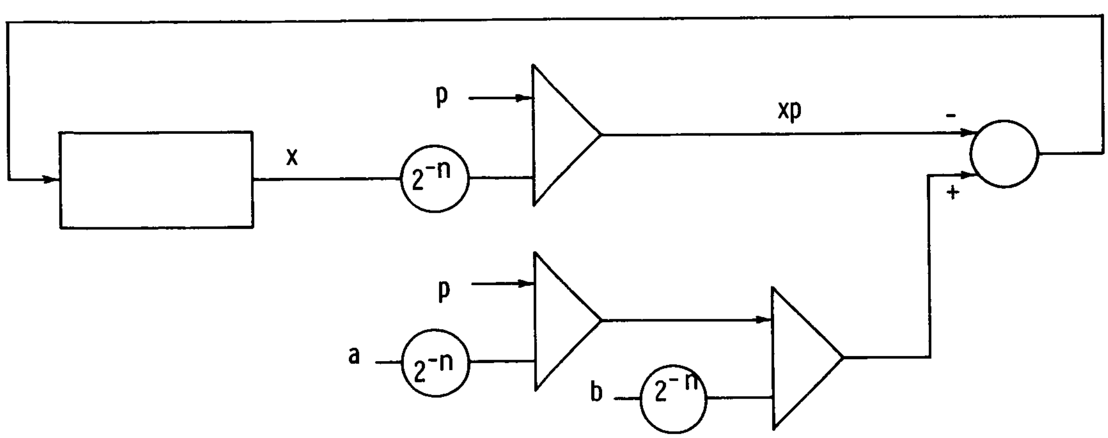


Figure 3.22. - Iterative process product machine.

$$x_{(k+1)} = x_{(k)} - 2^{-n}(a - x_{(k)})^2 \tag{3.63}$$

as the iterative process, then the process would not converge.

A <u>product</u> machine may also be designed using this method of synthesis. In particular, if

$$x = ab \tag{3.64}$$

then the product may be found by the iterative process

$$x_{(k+1)} = x_{(k)} - p(x_{(k)} - ab) \tag{3.65}$$

This machine is shown in Fig. 3.22. If the scale factors for x, a, and b are all taken as 2^n, then the machine generates the iterative process

$$x_{(k+1)} = x_{(k)} - 2^{-n}(x_{(k)} - ab) \tag{3.66}$$

The iterative truncation error for this machine in terms of $\mathcal{E}_{(0)}$ is

$$\mathcal{E}_{(k)} = (1 - 2^{-n})^k \mathcal{E}_{(0)} \tag{3.67}$$

Synthesis (Regenerative Circuit)

Consider the schematic diagram shown in Fig. 3.23. The value of K is bounded such that

$$|K| \leq 1 - 2^{-n} \tag{3.68}$$

The output equation for this circuit may be written as

$$dz = K(dx + dz)$$

$$\frac{dz}{dx} = \frac{K}{1 - K} \tag{3.69}$$

As $K \rightarrow 1$ in Eq. (3.69) then the ratio $dz/dx \rightarrow \infty$. However, Eq. (3.68) fixes an upper bound on this ratio such that

$$\frac{K}{1 - K} \cong 2^n - 1 \tag{3.70}$$

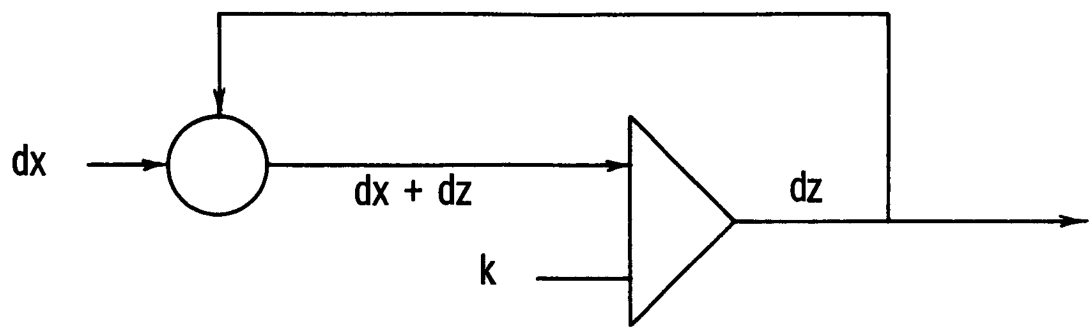


Figure 3.23. - Regenerative circuit.

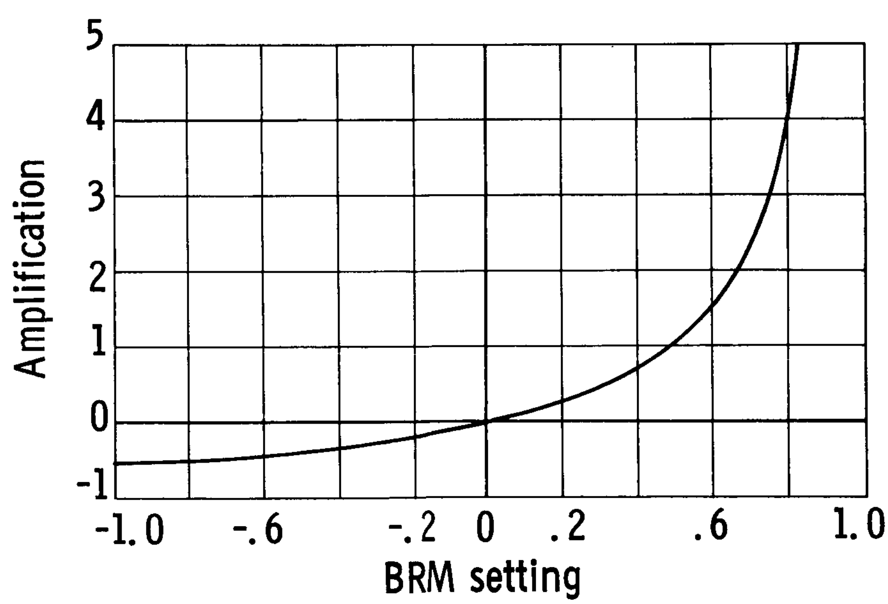


Figure 3.24. - Amplification of regeneration circuit.

Therefore, by using this regenerative circuit, the BRM may act as an amplifier. However, if large amplifications are to be considered other factors must be taken into account. In particular, suppose a dx pulse arrives at the BRM and this generates a dz pulse. The dz pulse is delayed by a gated pulse generator and is fed back to the BRM. This in turn may generate another dz pulse. This process may be continued depending on the value of K. However, each time a dz pulse is recirculated the pulse shape deteriorates. For example, if leading edge logic is used then the rise time of each pulse will be increased until the dz pulse is not sharp enough to be utilized. Moreover, enough time must be allowed between the dx pulses to permit the maximum number of dz pulses. The maximum value of K usually utilized in these circuits will in general be less than that permitted by Eq. (3.68).

This point is illustrated by the plot of Eq. (3.69) in Fig. 3.24. If $-1 < K \leq 1/2$ then at most 1 pulse will be fed back with each input pulse. If $1/2 < K < 1$ then more than 1 pulse will be fed back with each input pulse. Therefore, by fixing the upper value of K, the maximum number of feed-back pulses may be restricted.

The method of synthesis in this section is similar to that used in the synthesis by differential equation. However, in this section the differential equation will involve the highest order derivative on both sides of the equation; i.e.,

$$\frac{d^m y}{dx^m} = f\left(\frac{dy^m}{dx^m}, \; . \; . \; ., \; y, x\right) \tag{3.71}$$

In general the equation is written in this form when the highest order derivative can not be isolated In particular, if the highest order derivative has a non-constant coefficient then it may be written as Eq. (3.71) by adding and subtracting a constant from this coefficient (see Refs. 6 and 7). The design of the circuit based on Eq. (3.71) implies the use of the regenerative circuit since the generation of the highest order derivative involves itself.

Square Root Machine

Consider the generation of the square root

$$y = \sqrt{x} \tag{3.72}$$

from the equation

$$y \frac{dy}{dx} = 1/2 \tag{3.73}$$

The first step of this technique is to write Eq. (3.73) as

$$(y - C + C) \frac{dy}{dx} = 1/2 \tag{3.74}$$

and then isolate the highest order derivative as shown in the following equation.

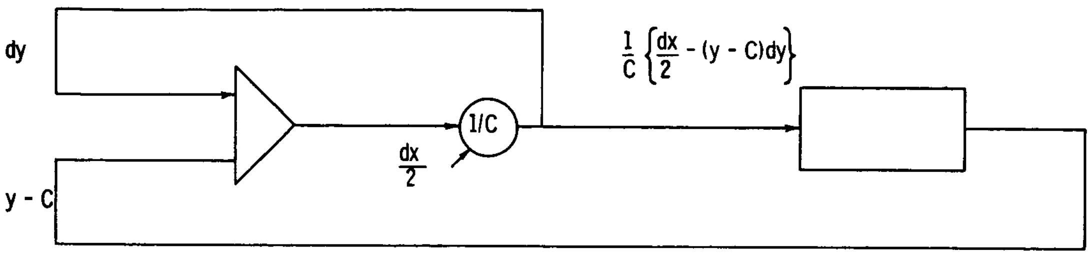
$$dy = \frac{1}{C} \left\{ \frac{dx}{2} - (y - C)dy \right\} \tag{3.75}$$

The basic and scaled circuit for generating Eq. (3.75) is shown in Figs. 3.25a and 3.25b, respectively. Following the same technique used to derive a machine based on a differential equation, the constraint equations are written.

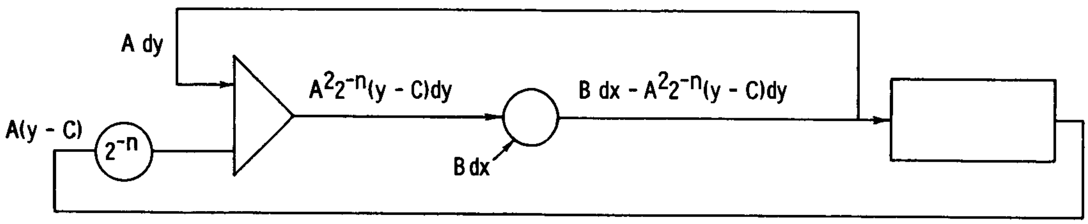
$$A\ dy = B\ dx - A^2 2^{-n}(y - C)dy \tag{3.76}$$

is the defining equation for the machine, and

$$A|C - y|_{max} \leq 2^n - 1 \tag{3.77}$$



(a) Basic design.



(b) Scaled design.

Figure 3.25. - Regenerative circuit square root machine.

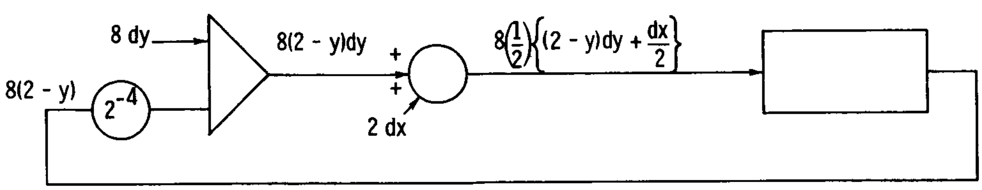


Figure 3.26. - Scaled diagram for four-stage square root machine.

is the counter limiting equation. From these equations the scale factors may be computed by

$$A = 2^n/C, \quad B = 2^{n-1}/C^2 \tag{3.78}$$

where C is chosen such that it satisfies the inequality

$$\left|1 - \frac{y}{C}\right|_{max} \leq 1 - 2^{-n} \tag{3.79}$$

Based on this design, Fig. 3.26 gives the computed scales and circuit for a four stage regenerative circuit square root machine. The output of this machine together with that of the desired output is shown in Fig. 3.27.

Other Regenerative Circuit Machines

The natural logarithm machine

$$y = \ln x \tag{3.80}$$

may be designed as a regenerative circuit by the equation

$$dy = \frac{dx}{C} + \left(1 - \frac{x}{C}\right) dy \tag{3.81}$$

The circuit for generating this function is shown in Fig. 3.28.

The quotient machine

$$z = x/y \tag{3.82}$$

may be designed as a regenerative circuit by the equation

$$dz = \frac{1}{a}\left[(y - a)\ dz + z\ dy - dx\right] \tag{3.83}$$

The schematic circuit for this function is shown in Fig. 3.29.

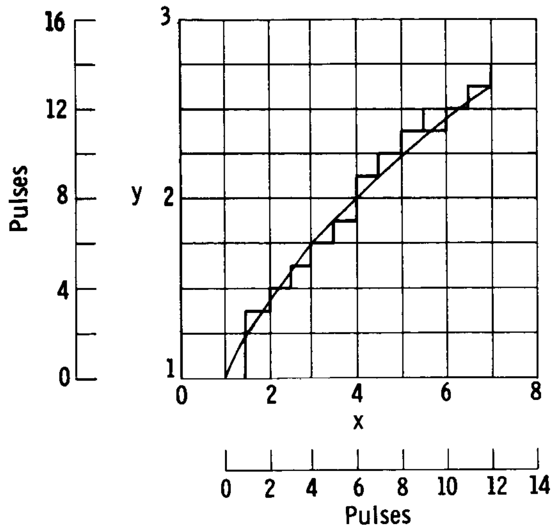


Figure 3.27. - Output of regenerative circuit square root machine.

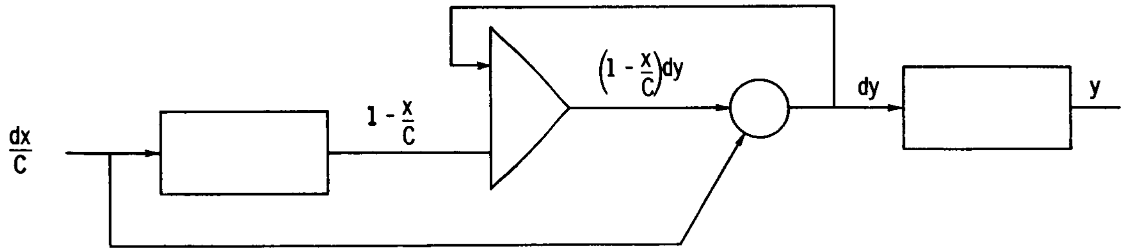


Figure 3.28. - Regenerative circuit logrithm machine.

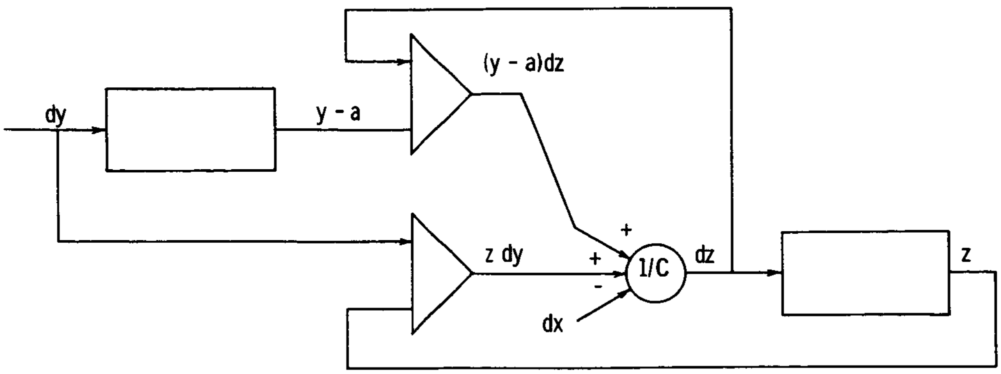


Figure 3.29. - Regenerative circuit quotient machine.

CHAPTER IV

PIECEWISE POLYNOMIAL MACHINES

Polynomial Machines

When a function to be generated is available only as empirical data (e.g., such as in sampled data systems) then the design must be based on generating an approximation function. Various classes of approximation functions and techniques for obtaining approximations have received considerable attention in the literature (e.g., see Ref. 14). A particularly convenient form for approximating a continuous function is that of a polynomial. The general polynomial

$$f(x) = \frac{a_m x^m}{m!} + \frac{a_{m-1}}{(m - 1)!} x^{m-1} + \ . \ . \ . + \frac{a_2}{2!} x^2 + a_1 x + a_0 \qquad (4.1)$$

may be generated by the circuit shown in Fig. 4.1. However, it is usually the case that all the data is not immediately available for generating the function over its entire range, or if it is available the polynomial needed to meet the accuracy requirements is of excessively high degree. In these applications the requirements of the problem may be met by using a series of relatively low degree polynomials where each polynomial is used to fit data only in a restricted range. Such machines are called piecewise polynomial machines.

Because of their wide spread use in applications (see Refs. 9,

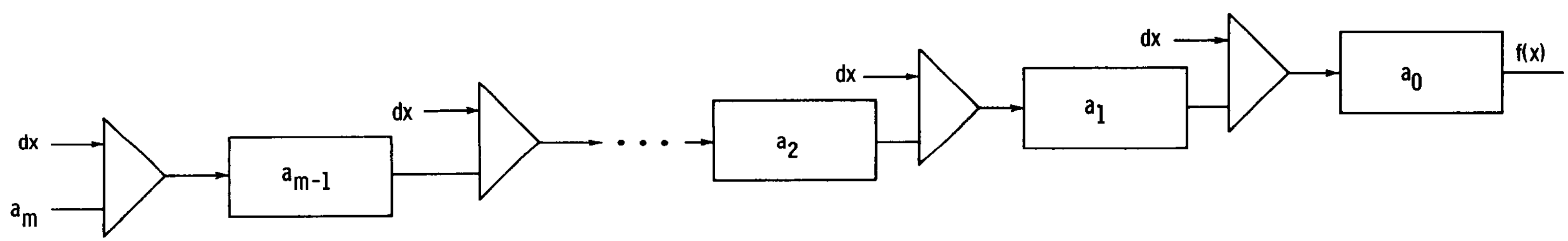


Figure 4.1. - General polynomial machine.

11, 18, and 19) and also because they can be realized with relatively simple circuits, this chapter will be devoted exclusively to describing a series of machines for generating piecewise polynomials arising from finite difference techniques. These machines are grouped into two broad categories based on applications; i.e., interpolation or extrapolation. Each machine of this series will generate a low order polynomial fitted to data available at equal intervals of the argument. In passing from one segment into the next new data is introduced. The form of the data in each case is simply generated from the empirical data.

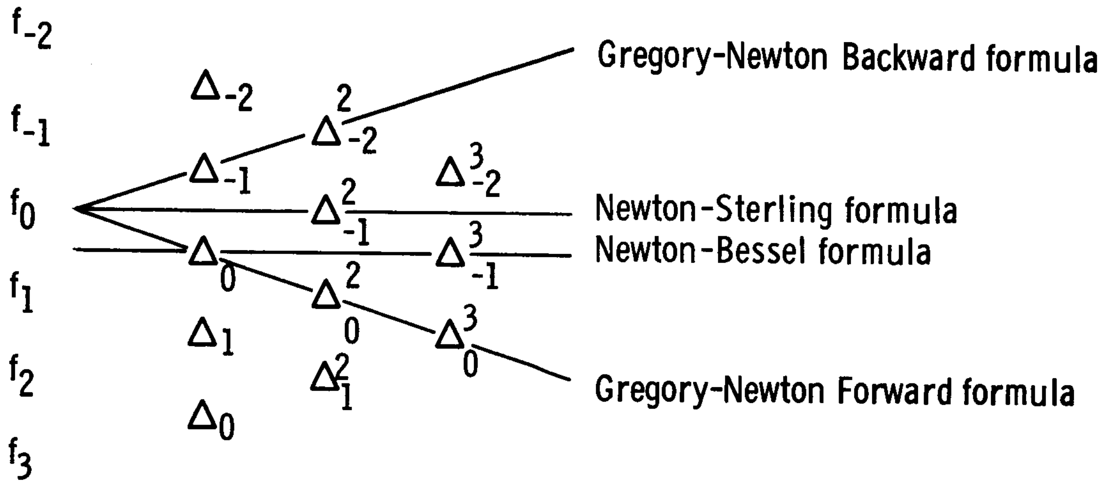
In order to facilitate the description of the machines in the next two sections, ordinary difference notation will be used. In particular, ω_n is the value of the independent variable where the value of the function is obtained; i.e.,

$$f(\omega_n) = f_n \tag{4.2}$$

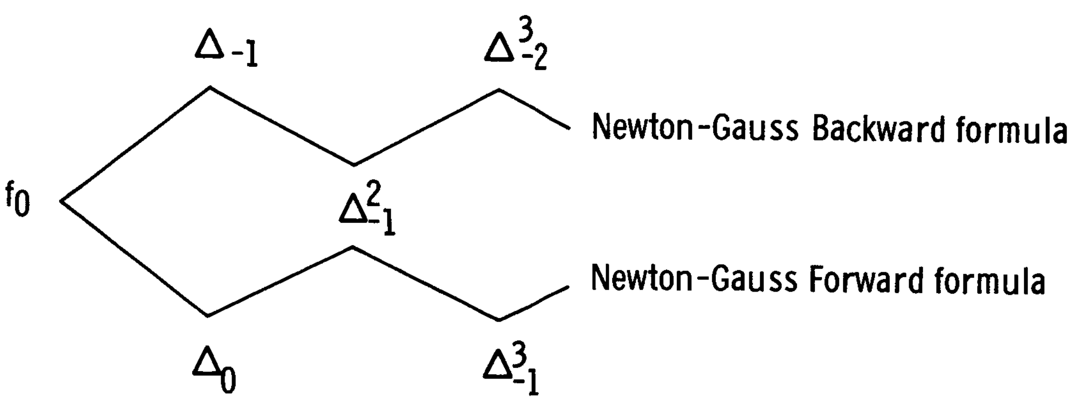
The quantity $\delta\omega$ represents the spacing of the independent variable, and $\Delta_n, \Delta_n^2, \ldots, \Delta_n^m$ are the successive differences which may be obtained from lower order differences as follows:

$$\begin{aligned} \Delta_n &= f_{n+1} - f_n \\ \Delta_n^2 &= \Delta_{n+1} - \Delta_n \\ &\vdots \\ \Delta_n^m &= \Delta_{n+1}^{m-1} - \Delta_n^{m-1} \end{aligned} \tag{4.3}$$

In formulating the approximation formula which passes through the given points, it is convenient to display these various differences in tabular form as shown in Fig. 4.2. From this difference



(a) Direct path difference formulas.



(b) Broken path difference formulas.

Figure 4. 2. - Difference table and paths of difference formulas.

table, alternate forms of the approximation formula may be derived dependent on the differences which are utilized; i.e., on the path through the difference table. The paths of some of these formulas are shown in Fig. 4.2. The form of the formula which will be utilized in the following discussion will be such that in each case the value of the variable x will vary from 0 to 1 in the interval of interest.

Interpolation

A piecewise linear interpolator can be obtained by passing a linear polynomial through successive pair of points of the function to be generated. This scheme is illustrated in Fig. 4.3. A linear polynomial is generated which passes through the points P_0 and P_1. At the point P_1, the point P_2 is added to the scheme and a linear polynomial is generated which passes through the points P_1 and P_2. This procedure may be expressed in terms of ordinary differences by the Gregory-Newton interpolation formula; i.e.,

$$f(\omega_n + x\delta\omega) = f_n + \Delta_n x \tag{4.4}$$

The first derivative of this formula is

$$\delta\omega f'(\omega_n + x\delta\omega) = \Delta_n \tag{4.5}$$

This interpolation formula may be generated by the linear polynomial generator shown in Fig. 4.4a.

The corresponding equation and its first derivative for the next interpolation interval is given by,

$$f(\omega_{n+1} + x\delta\omega) = f_{n+1} + \Delta_{n+1} x \tag{4.6}$$

$$\delta\omega f'(\omega_{n+1} + x\delta\omega) = \Delta_{n+1} \tag{4.7}$$

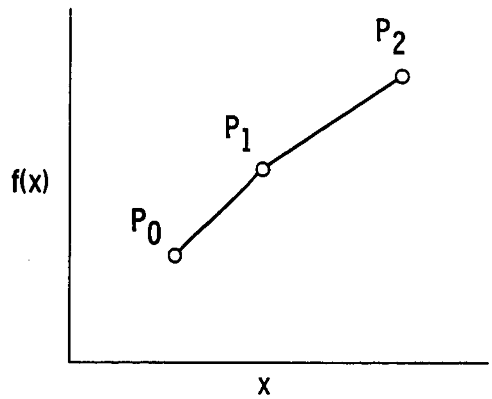
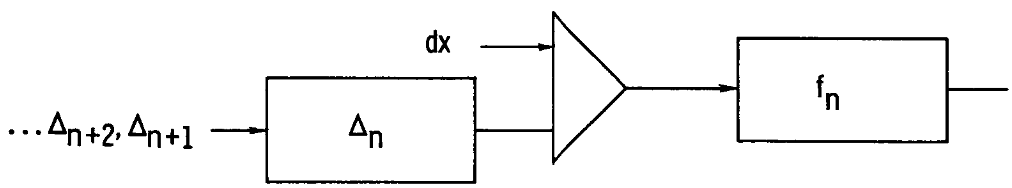
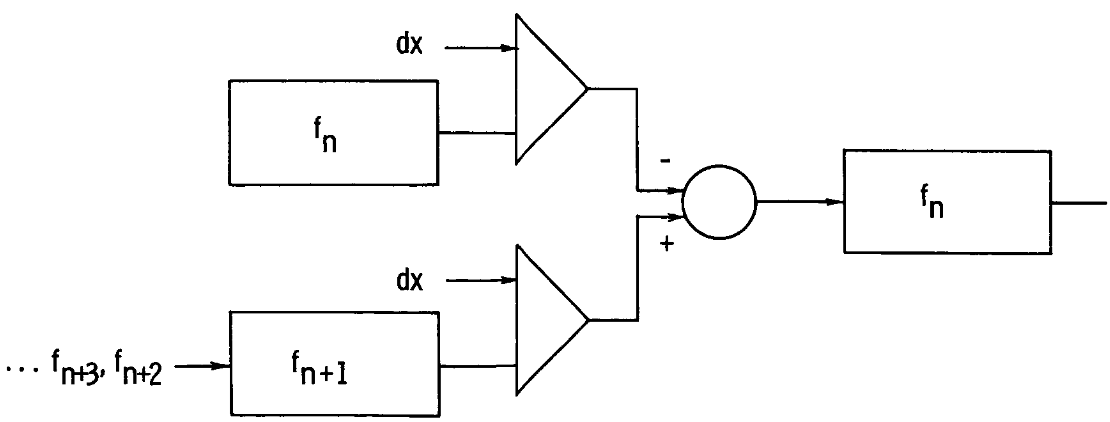


Figure 4.3. - Scheme for piecewise linear interpolation.



(a) First difference input data.



(b) Function values input data.

Figure 4.4. - Machines for piecewise linear interpolation.

At the end of the first interval of the values of the function and its derivative given by equations (4.4) and (4.5) are

$$f(\omega_n + \delta\omega) = f_n + \Delta_n = f_{n+1} \tag{4.8}$$

$$\delta\omega f'(\omega_n + \delta\omega) = \Delta_n \tag{4.9}$$

The corresponding values of these two quantities needed at the start of the next interval are given by Eqs. (4.6) and (4.7) and are:

$$f(\omega_{n+1}) = f_{n+1} \tag{4.10}$$

$$\delta\omega f'(\omega_{n+1}) = \Delta_{n+1} \tag{4.11}$$

By direct computation it may be verified that in order to proceed from one interval to the next, the quantity Δ_n^2 (which is $\Delta_{n+1} - \Delta n$) needs to be added to the setting of the BRM, and the output (i.e., the end point of the interpolation interval) need not be modified. However, since adders have been excluded as basic design elements, the same result may be attained by transferring Δ_{n+1} as the setting of the BRM (since $\Delta_{n+1} = \Delta_n + \Delta_n^2$). Consequently, the circuit shown in Fig. 4.4a may be used for piecewise linear interpolation of a function by transferring successive first difference as settings for the BRM in order to proceed from one interval to the next.

The circuit derived above is well suited for an application in which an incremental encoder is used to generate the input data. If an absolute encoder is used to generate the primary data, then the above circuit may be adapted for this input by using the defining equation for first differences; i.e., Eq. (4.3). This circuit is shown in Fig. 4.4b. As in the previous case, only one new piece of information must be transferred into the circuit in order to proceed from one interpolation interval to the next. However, in this case, before the new information, i.e., f_{n+2}, is transferred as the setting of the lower BRM, the value of the lower BRM; i.e., f_{n+1}, must be transferred to the upper one.

Two piecewise quadratic interpolators will be derived. The scheme for the first one which will be called the back interval quadratic interpolator is illustrated in Fig. 4.5. A quadratic polynomial is generated which passes through points P_0, P_1, and P_2. This polynomial is used to generate the curve between points P_0 and P_1. The point P_3 is then added to the scheme and a quadratic polynomial is derived which passes through the points P_1, P_2, and P_3. This polynomial is then used to generate the curve between P_1 and P_2.

This procedure may also be conveniently expressed by the Gregory-Newton quadratic interpolation formula; i.e.,

$$f(\omega_n + x\delta\omega) = f_n + \Delta_n x + \frac{x(x-1)}{2} \Delta_n^2 \tag{4.12}$$

The successive derivatives for this formula are

$$\delta\omega f'(\omega_n + x\delta\omega) = (\Delta_n - \frac{1}{2}\Delta_n^2) + \Delta_n^2 x \tag{4.13}$$

$$(\delta\omega)^2 f''(\omega_n + x\delta\omega) = \Delta_n^2 \tag{4.14}$$

The corresponding equation and its derivatives for the next interpolation interval are:

$$f(\omega_{n+1} + x\delta\omega) = f_{n+1} + (\Delta_{n+1} - \frac{1}{2}\Delta_{n+1}^2)x + \frac{\Delta_{n+1}}{2} x^2 \tag{4.15}$$

$$\delta\omega f'(\omega_{n+1} + x\delta\omega) = \Delta_{n+1} - \frac{1}{2}\Delta_{n+1}^2 + \Delta_{n+1}^2 x \tag{4.16}$$

$$(\delta\omega)^2 f''(\omega_{n+1} + x\delta\omega) = \Delta_{n+1}^2 \tag{4.17}$$

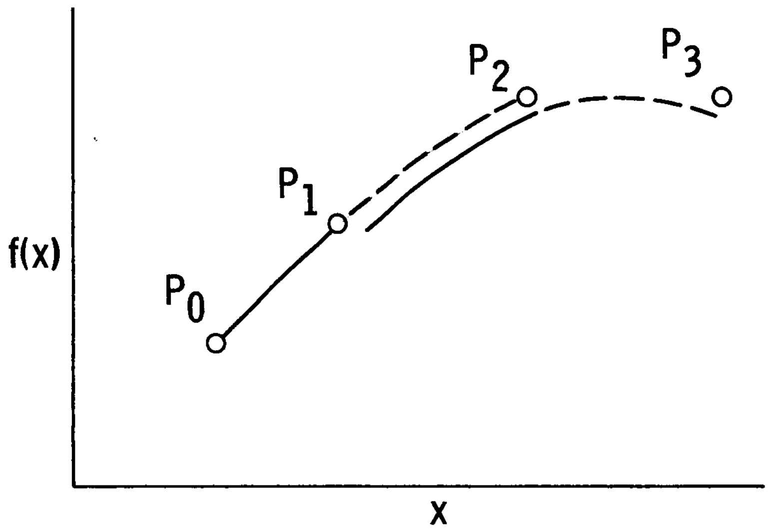


Figure 4.5. - Scheme for piecewise quadratic interpolation (back interval).

Consequently, the correction to be added to the second derivative, first derivative, and function at the end of the interval in order to proceed to the next interval are Δ^3_{n+1}, $-\frac{1}{2}\Delta^3_{n+1}$, and 0, respectively. From this formulation, however, addition is required in order to proceed from one interval into the next. A formulation of this process which leads to the elimination of the explicit adder is to splinter the polynomial given by Eq. (4.12) into the following two polynomials.

$$f_a(\omega_n + x\delta\omega) = f_n + \Delta_n x + \frac{\Delta^2_n}{2} x^2 \tag{4.18}$$

$$f_b(\omega_n + x\delta\omega) = -\frac{\Delta^2_n}{2} x \tag{4.19}$$

where

$$f(\omega_n + x\delta\omega) = f_a(\omega_n + x\delta\omega) + f_b(\omega_n + x\delta\omega) \tag{4.20}$$

The first and second derivatives of Eq. (4.18) are

$$(\delta\omega)f_a'(\omega_n + x\delta\omega) = \Delta_n + \Delta^2_n x \tag{4.21}$$

$$(\delta\omega)^2 f_a''(\omega_n + x\delta\omega) = \Delta^2_n \tag{4.22}$$

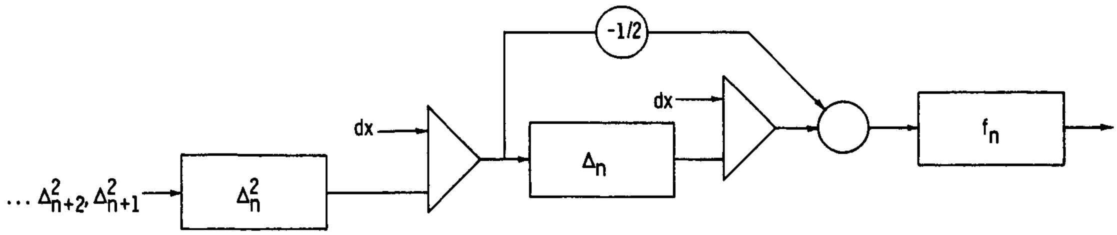
The corresponding splintering of Eq. (4.15) yields

$$f_a(\omega_{n+1} + x\delta\omega) = f_{n+1} + \Delta_{n+1} x + \frac{\Delta^2_{n+1}}{2} x^2 \tag{4.23}$$

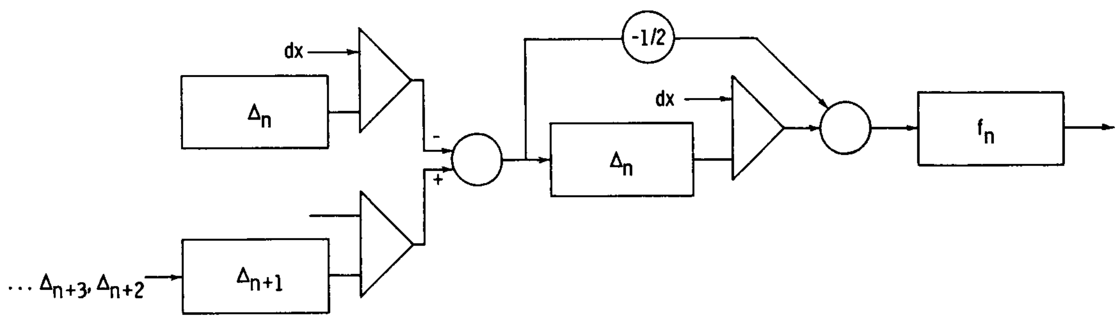
$$(\delta\omega)f_a'(\omega_{n+1} + x\delta\omega) = \Delta_{n+1} + \Delta^2_{n+1} x \tag{4.24}$$

$$(\delta\omega)^2 f_a''(\omega_{n+1} + x\delta\omega) = \Delta^2_{n+1} \tag{4.25}$$

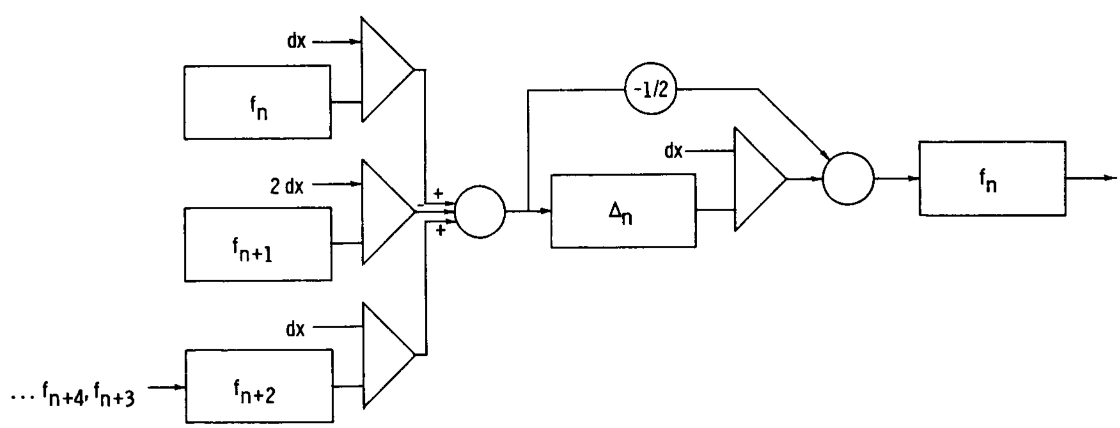
Consequently, no correction need be adder to the first derivative in generating the function f_a. Eqs. (4.18) and (4.19) may then be used to design the circuit shown in Fig. 4.6a. It will be noted that in order to proceed from one interval to the next, only



(a) Second difference input data.



(b) First difference input data.



(c) Function values input data.

Figure 4.6. - Machines for piecewise quadratic interpolation (back interval).

new second difference data need be transferred to set the levels of the leftmost BRM.

Based on the defining equation for second differences; i.e., Eq. (4.3), the machines shown in Figs. 4.6b and 4.6c may be obtained directly from the machine shown in Fig. 4.6a. In these machines, previous first differences and values of the function are transferred directly from the lower BRM's to the upper ones before a new first difference and a value of the function, respectively, is transferred into the lower one in order to proceed from one interval into the next.

The scheme for piecewise front interval quadratic interpolation illustrated in Fig. 4.7 may be derived by use of the Newton-Gauss interpolation formula given in the following equation.

$$f(\omega_n + x\delta\omega) = f_n + x\,\Delta_{n-1} + \frac{x(x+1)}{2}\,\Delta^2_{n-1} \tag{4.26}$$

If Eq. (4.26) is implemented directly, then the first derivative and second derivative must be corrected by adding $\frac{1}{2}\Delta^3_{-1}$ and Δ^3_{-1}, respectively, to these quantities in order to proceed from one interval to the next. The explicit need for an adder may be avoided in a manner similar to that used in the previous discussion by splintering Eq. (4.26) into the following pair of equations.

$$f_a(\omega_n + x\delta\omega) = f_n + x\,\Delta_{n-1} + \frac{1}{2}\,x^2\,\Delta^2_{n-1} \tag{4.27}$$

$$f_b(\omega_n + x\delta\omega) = \frac{1}{2}\,\Delta^2_{n-1}\,x \tag{4.28}$$

Based on this pair of equations, the circuit shown in Fig. 4.8a may be derived directly. The machines shown in Figs. 4.8b and 4.8c

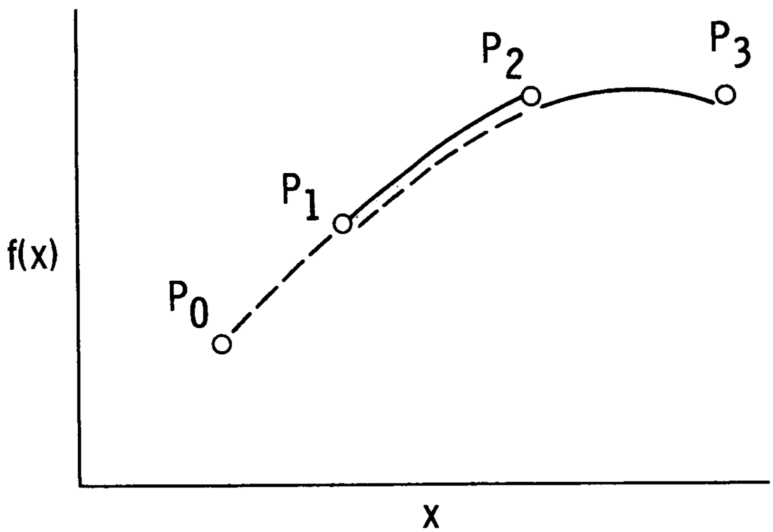
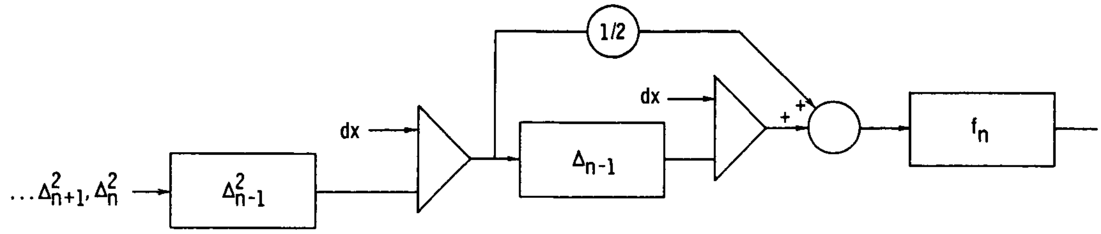
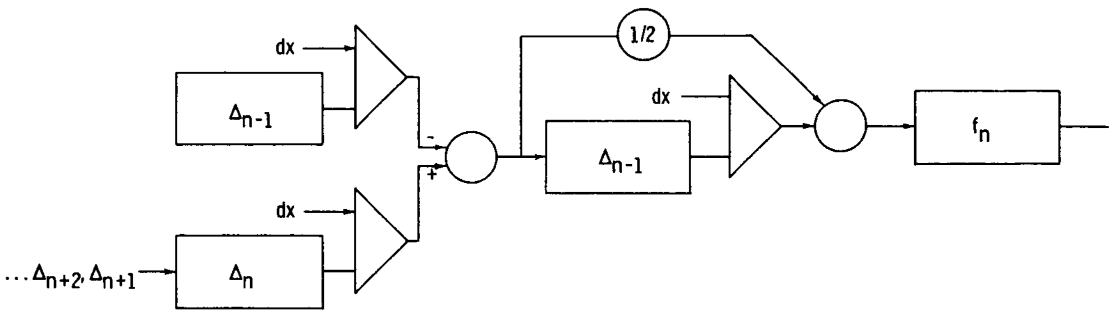


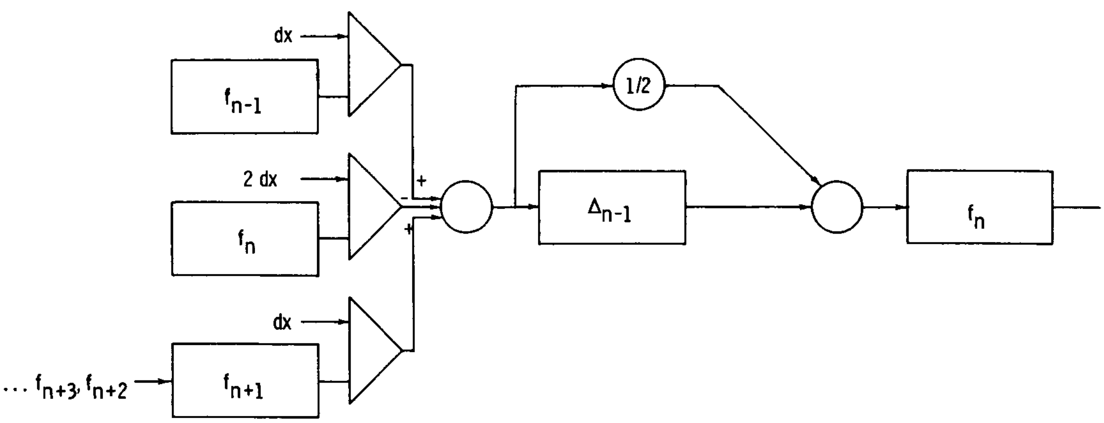
Figure 4.7. - Scheme for piecewise quadratic interpolation (front interval).



(a) Second difference input data.



(b) First difference input data.



(c) Function values input data.

Figure 4.8. - Machines for piecewise quadratic interpolation (front interval).

are adapted from the circuit shown in Fig. 4.8a by using the definition of the second difference given in Eq. (4.3).

A cubic interpolator may be obtained from the Newton-Gauss interpolation; i.e.,

$$f(\omega_n + x\delta\omega) = f_n + x\,\Delta_n + \frac{1}{2}\,x(x-1)\,\Delta^2_{n-1} + \frac{1}{6}\,x(x-1)(x+1)\,\Delta^3_{n-1} \tag{4.29}$$

However, in this case the discussion will be limited to using this formula for central interval interpolation only. This scheme is illustrated in Fig. 4.9. The points P_0, P_1, P_2, and P_3 are used to generate an interpolation formula for interpolating between P_1 and P_2. The point P_4 is then added to the scheme and the points P_1, P_2, P_3, and P_4 are used to interpolate between points P_2 and P_3.

Eq. (4.29) may be applied directly to yield a central interval cubic interpolator. However, in this case the third, second, and first derivatives must be corrected by adding Δ^4_{-1}, 0, and $\frac{1}{2}\,\Delta^4_{-1}$ to these quantities, respectively, in order to proceed from one interval to the next.

A configuration may be obtained which conforms with the design practice of not using an adder by splintering Eq. (4.29) into the following pair of equations.

$$f_a(\omega_n + x\delta\omega) = f_n + x\left(\Delta_n - \frac{1}{2}\,\Delta^2_{n-1}\right) + \frac{x^2}{2}\,\Delta^2_{n-1} + \frac{x^3}{6}\,\Delta^3_{n-1} \tag{4.30}$$

$$f_b(\omega_n + x\delta\omega) = -\frac{x}{6}\,\Delta^3_{n-1} \tag{4.31}$$

Based on this pair of equations, a circuit may be obtained such that the function, its first derivative, and its second derivative

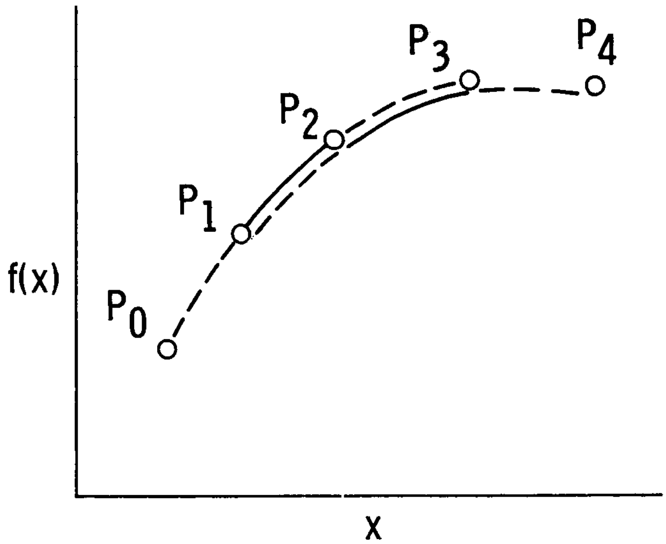


Figure 4.9. - Scheme for piecewise cubic interpolation (central interval).

need not be changed in order to proceed from one interval to the next. This circuit is shown in Fig. 4.10a. The circuits presented in Figs. 4.10b, 4.10c, and 4.10d are modifications of this circuit based on the definition of the third difference.

Extrapolation

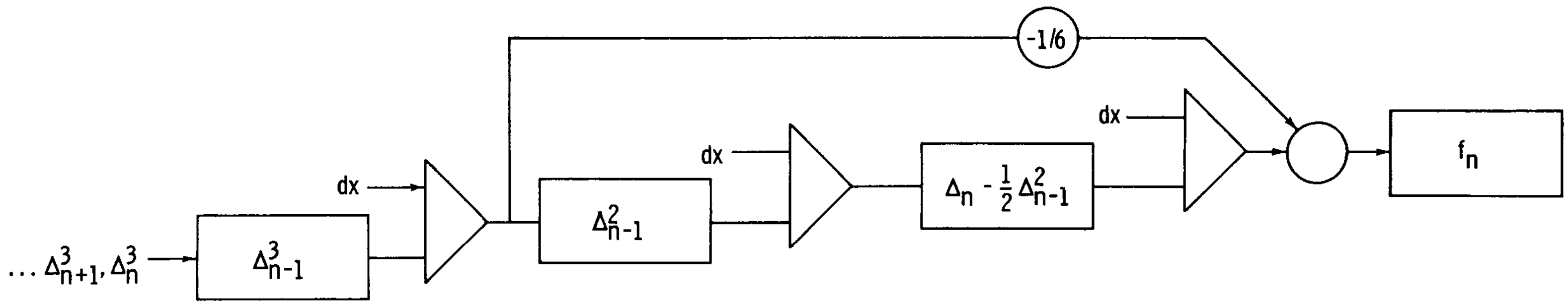
Extrapolation presents an added problem in that the output of the machine must also be corrected in order to proceed from one interval to the next. This is illustrated in Fig. 4.11 for linear extrapolation. A linear polynomial through P_0 and P_1 is used to extrapolated the values from P_1 to P_2^*. The point P_2 is then added to the scheme, and a linear polynomial through P_1 and P_2 is used to extrapolate the next interval. The predicted value P_2^* and the new value P_2 can be expected to be different. Consequently, the output must be corrected for this new value P_2. In order to avoid putting a jump in the output function at this point, the scheme which will be employed is to put the correction in linearly over the entire next interval. This scheme (as well as that of quadratic extrapolation which will be described next) is closely related to the Porter-Stoneman digital filters (see Ref. 13) and may be extended accordingly.

The Gregory-Newton backward finite difference formula may be used to design the <u>linear extrapolation machine</u>; i.e.,

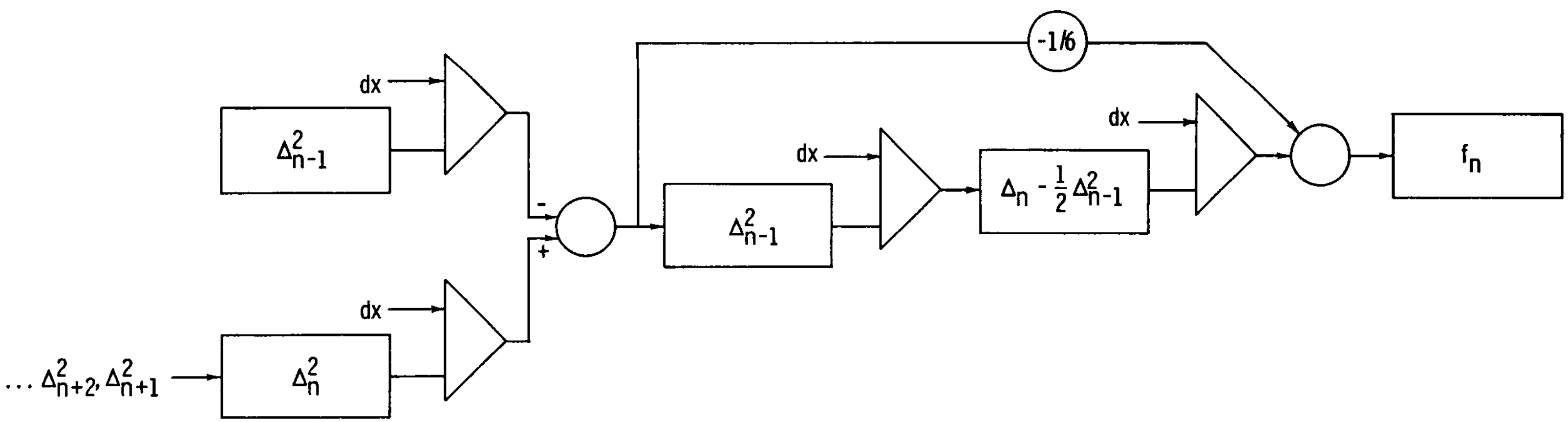
$$f(\omega_n + x\delta\omega) = f_n + \Delta_{n-1}x \tag{4.32}$$

The corresponding formula for extrapolating the next interval is

$$f(\omega_{n+1} + x\delta\omega) = f_{n+1} + \Delta_n x \tag{4.33}$$

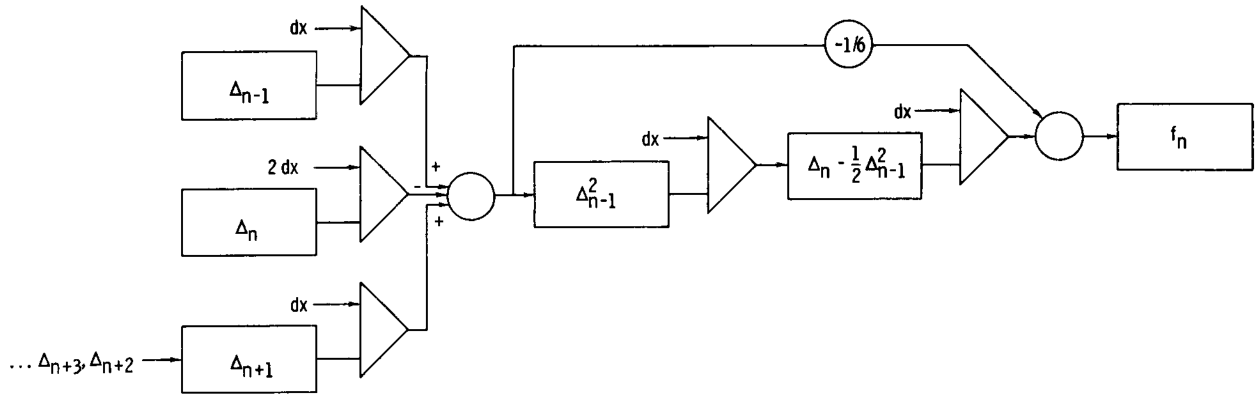


(a) Third difference input data.

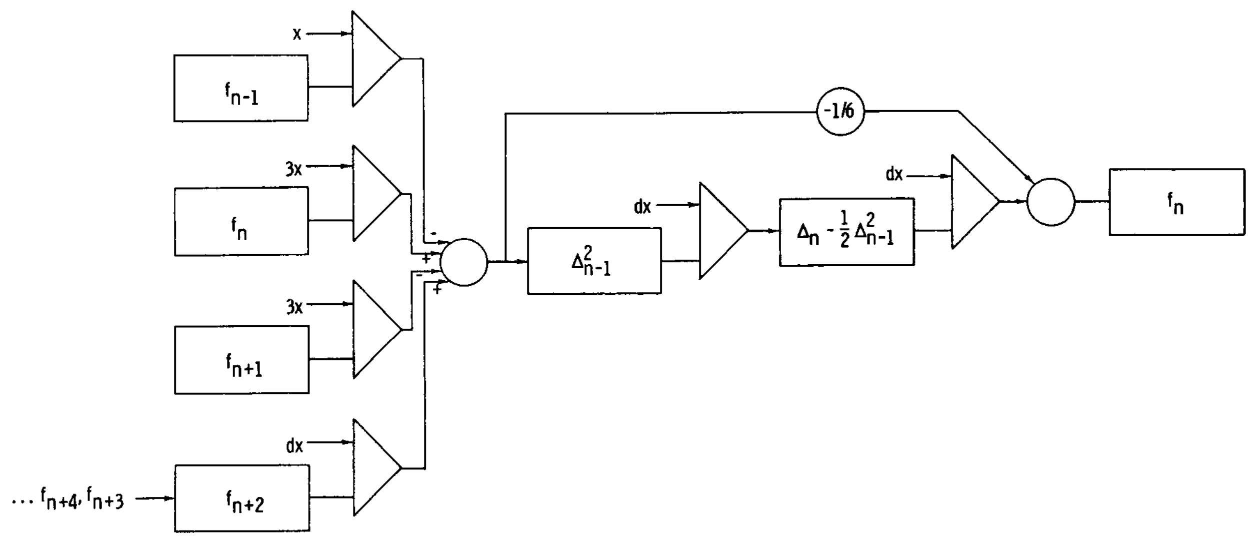


(b) Second difference input data.

Figure 4.10. - Machines for piecewise cubic interpolation (central interval).



(c) First difference input data.



(d) Function values input data.

Figure 4.10. - Continued. Machines for piecewise cubic interpolation (central interval).

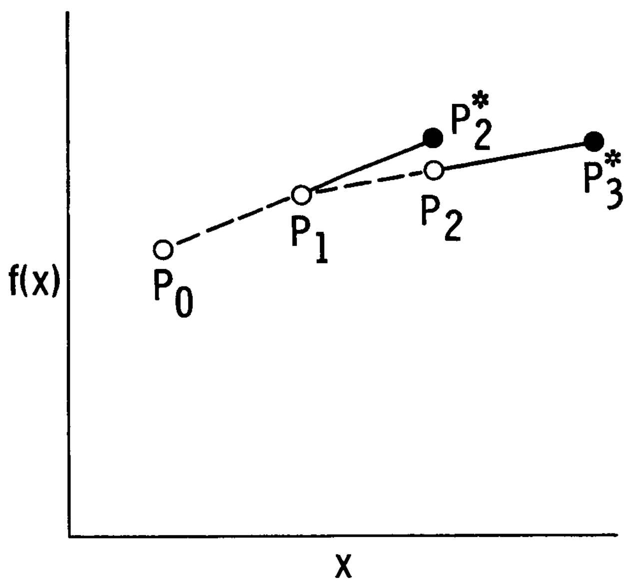


Figure 4.11. - Scheme for piecewise linear extrapolation.

If these formulas are applied directly, then the circuit must be corrected at the end of the interval by adding Δ^2_{n-1} to both the function and its first derivative in order to proceed into the next interval. This, however, would cause a jump in the output function. This jump may be avoided by putting the correction term in the output in a linear manner over the entire next interval. The resulting polynomial has the property that its initial value corresponds to the end point of Eq. (4.32) and its final value corresponds to the end point of Eq. (4.33). A polynomial which satisfied these constraints may be written as follows:

$$f(\omega_{n+1} + x\delta\omega) = (f_n + \Delta_{n-1}) + (\Delta_n + \Delta^2_{n-1})x \tag{4.34}$$

The second difference in Eq. (4.34) may be eliminated by using the defining equation given by Eq. (4.3). This substitution yields the following equivalent equation.

$$f(\omega_{n+1} + x\delta\omega) = (f_n + \Delta_{n-1}) + (2\Delta_n - \Delta_{n-1})x \tag{4.35}$$

Eq. (4.35) may be implemented to yield the linear extrapolator shown in Fig. 4.12.

The scheme for quadratic extrapolation is shown in Fig. 4.13. The quadratic equation through points P_0, P_1, and P_2 is used to extrapolate the data to the point P_3^*. The point P_3 is then added to the scheme and can in general be expected to be different from P_3^*. The quadratic equation through the points P_1, P_2, and P_3 is then used to extrapolate to the point P_4^*. In order to avoid putting a jump in the output when new information is added

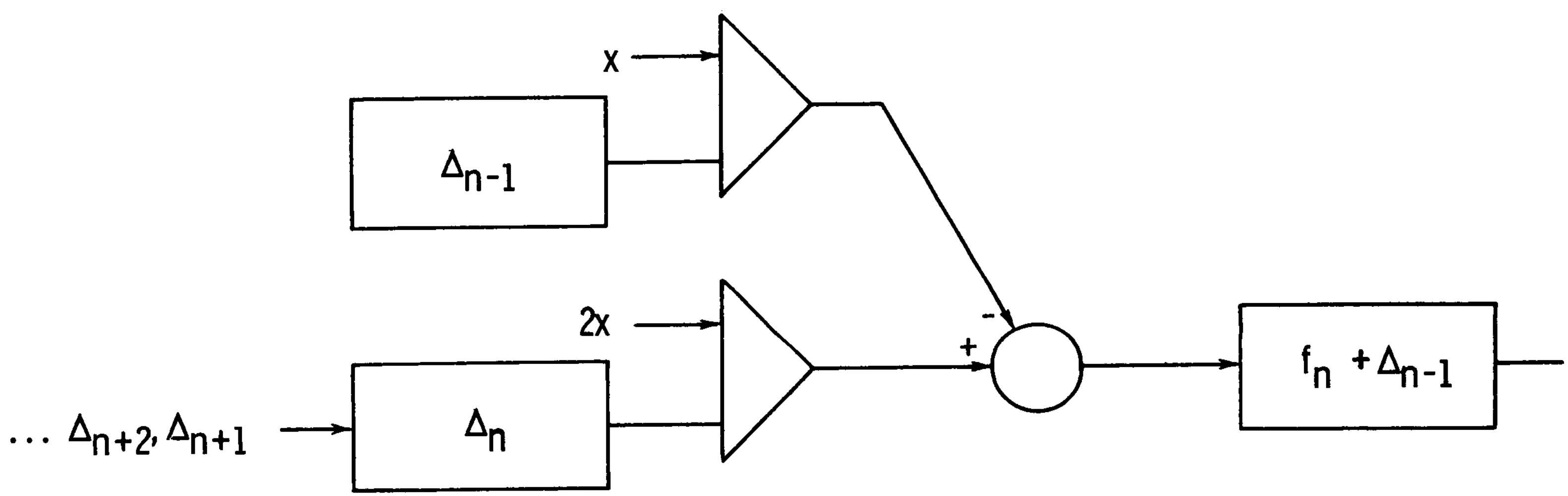


Figure 4.12. - Machine for piecewise linear extrapolator.

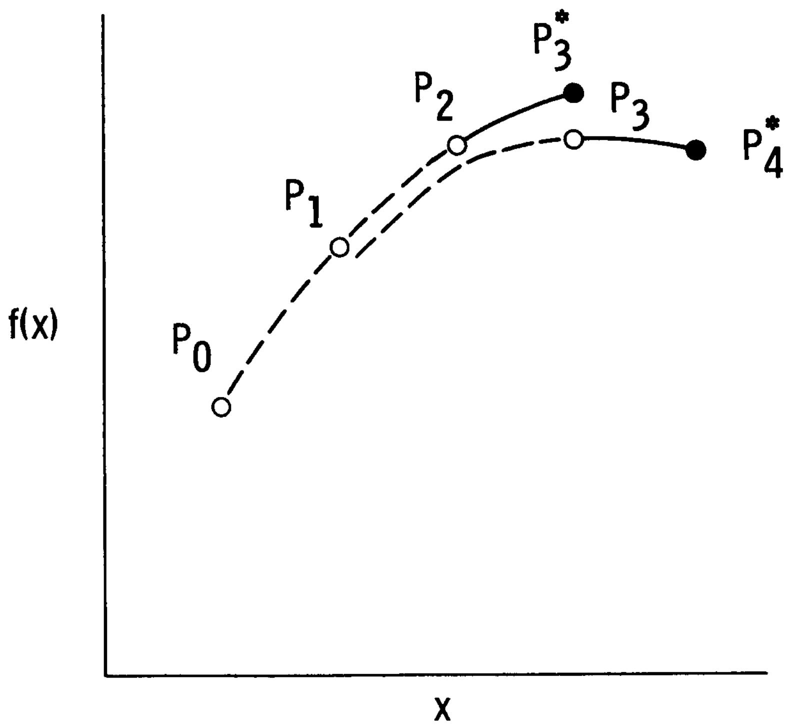


Figure 4.13. - Scheme for piecewise quadratic extrapolation.

to the scheme, the correction may be put into the output in a linear manner over the entire next interval in the same manner as that employed for linear extrapolation.

The Gregory-Newton backward difference formula forms the basis of the <u>quadratic</u> extrapolation. This formula may be written as follows:

$$f(\omega_n + x\delta\omega) = f_n + x\,\Delta_{n-1} + \frac{x(x+1)}{2}\Delta^2_{n-2} \tag{4.36}$$

The corresponding formula for the next interval is:

$$f(\omega_{n+1} + x\delta\omega) = f_{n+1} + x\,\Delta_n + \frac{x(x+1)}{2}\Delta_{n-1} \tag{4.37}$$

If Eqs. (4.36) and (4.37) are implemented directly then the quantities Δ^3_{n-2}, $3\Delta^3_{n-2}/2$, and Δ^3_{n-2} must be added to the output function, its derivative, and its second derivative in order to proceed from one interval to the other. As was indicated earlier, the jump in the output function can be avoided by putting in the correction over the entire next interval. A polynomial which satisfies these constraints (i.e., has the end of Eq. (4.36) as its initial point and the end of Eq. (4.37) as its final point) may be written as follows:

$$f(\omega_{n+1} + x\delta\omega) = f_n + \Delta_{n-1} + \Delta^2_{n-2} + \left(\Delta_n + \frac{\Delta^2_{n-1}}{2} + \Delta^3_{n-2}\right)x + \frac{\Delta^2_{n-1}}{2}\,x \tag{4.38}$$

Substituting the difference relationship given by Eq. (4.3) into Eq. (4.38) yields

$$f(\omega_{n+1} + x\delta\omega) = f_n + \Delta_{n-1} + \Delta^2_{n-2}$$

$$+ \left(\Delta_{n-1} + \frac{5}{2}\Delta^2_{n-1} - \Delta^2_{n-2}\right)x + \frac{\Delta^2_{n-1}}{2}x^2 \qquad (4.39)$$

Eq. (4.39) may be splintered into the two equation

$$f_a(\omega_{n+1} + x\delta\omega) = f_n + \Delta_{n-1} + \Delta^2_{n-2} + \Delta_{n-1}x + \frac{\Delta^2_{n-1}}{2}x^2 \qquad (4.40)$$

$$f_b(\omega_{n+1} + x\delta\omega) = \left(\frac{5}{2}\Delta^2_{n-1} - \Delta^2_{n-2}\right)x \qquad (4.41)$$

to yield the circuit shown in Fig. 4.14.

The circuits shown in Figs. 4.12 and 4.14 may be readily expanded by use of finite difference relations (as was done for interpolators) to yield circuits which accept functional values and first differences as the primary source of data.

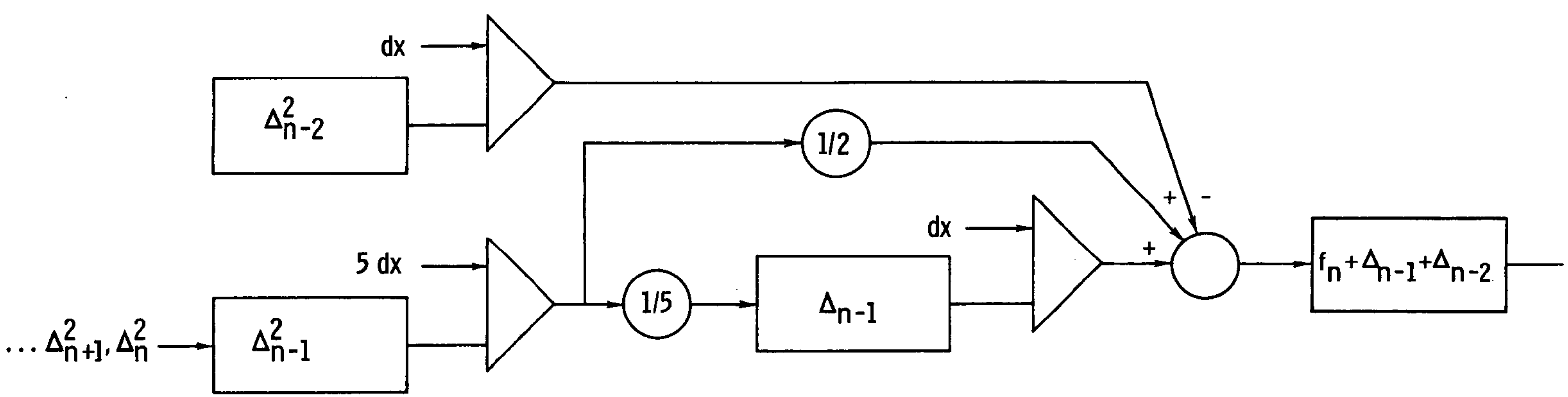


Figure 4.14. - Machine for piecewise quadratic extrapolator.

CHAPTER V

CONCLUSION

Summary

A handful of circuits have been reported in the literature which have been designed to meet the needs of special purpose digital computer problems arising from real time applications. The organization of these circuits is to utilize simple counting techniques as the basis for computing and result in a simplicity of hardware which make them attractive for such special purpose applications. The design of these circuits have been examined in this study with the objective of (1) "explaining" the circuits and (2) generalizing the design philosophy such that new circuits may be admitted with the same organization. In order to be specific we limited the principle design elements to three fundamental units. The elements are (1) the binary rate multiplier which is a means of scaling down a pulse stream to some specified fraction, (2) the counter, and (3) the anti-coincidence circuit which is a means of separating pulses arriving at a counter simultaneously.

These design elements are represented as operational units which may be used to describe the machines. Operational techniques are then used as the method of synthesis. In particular, a counter is utilized to represent a first order difference equation and a counter in cascade with a BRM is utilized to represent approximate integration. The computational errors; i.e., rounding-off error and truncation error, introduced into the machines as a result of treating the principle design elements as operational units are identified and studied in detail. The rule of round-off, which is simply stated in conventional computers, is not as easily formulated in these machines. Definite results, however, were obtained and the rounding-off error was shown to be dependent on the starting value of the BRM counter as well as n, the number of stages. The approximate error bound of $7/9 + n/3$ for the generated round-off error proved to be disappointingly pessimistic for predicting the propagated error for design purposes. Nevertheless, having identified these two sources are errors permitted us to obtain better results experimentally by two methods; (1) increasing the number of stages and (2) changing the round-off error by changing the starting value of the BRM counter.

The method of synthesis is presented in three parts; (1) expressing the function to be generated as a differential equation, (2) expressing it as the fixed point of an iterative process, and (3) expressing it in terms of a regenerative circuit which is presented. The method of synthesis is explicitly stated and is satisfactory in that all known circuits may be directly obtained from it. A wide variety of other functions are also obtained using these synthesis techniques. Many of these examples are illustrated and in some cases actual experimental results were obtained and discussed with the machine.

A series of machines is presented for interpolation and extrapolation of a function which is available only as empirical data. In particular, the function is generated over its entire range by a sequence of low order polynomials. Finite difference techniquss are used to describe the polynomials. The order of the polynomial is limited to a cubic for interpolation, and a quadratic for extrapolation since these seem to be the important cases in practice. Nevertheless, these techniques can be easily extended to include higher order polynomials.

Recommendations for Further Investigations

We feed that the choice of principle design elements has been correctly limited to units that operate as incremental devices. It would be interesting to investigate other components in this framework. In selecting the new components two approaches appear apparent. First, the components used in this study may be subdivided into smaller functional units with a view of studying simplification methods of the final design. Secondly, new functional units may be introduced with a view of admitting new machines. However, if components which operate on the whole word are included they should be simple decision type circuits (e.g., sign and magnitude comparators) and not new fundamental units like an adder which would dominate or supplant the other components.

It is expected that using worst case conditions for obtaining error bounds would not produce satisfactory design results which may be used in the whole spectrum of problems. The study which should produce good results would be to consider each circuit individually and obtain deterministic design results which can be applied to that circuit. In particular, an algebraic approach as was used in Appendices A and B might yield satisfactory results for processes which involve only addition and multiplication such as the generation of polynomials. For transcendental functions the bounds may be obtained by experimental techniques or perhaps by comparing the desired function to one which is attainable by algebraic means.

New methods of synthesis should also be sought either to include the pathological cases discussed earlier in the report or to exclude them as possible machines.

Other piecewise curve fitting machines should also be stuided, especially those in which functions other than low order polynomials are used and those in which the first and higher order derivatives are kept continuous.

The investigation of these machines would be facilitated if hardware and good display facilities were available which would permit the circuit to be easily fabricated and studied. We do not have in mind the design of still another general purpose computer since it is felt that these circuits best serve the needs of special purpose applications.

REFERENCES

1. von Neumann, S., and Goldstine, H. H.: "Numerical Inverting of Matrices of High Order", Bulletin of Am. Math. Soc., vol. 53, no. 11 (Nov. 1947), pp. 1021-1099.

2. Shileiko, A. V.: "Digital Models", Avtomatika i Telemekhanika, vol. 20 (Dec. 1959), pp. 1638-1649.

3. Harris, J. N.: "A Programmed Variable-Rate Counter for Generating the Sine Function", Trans. IRE PGEC, EC-5, no. 1 (March, 1956).

4. Gordon, B. M.: "Adapting Digital Techniques for Automatic Controls - I", Electrical Mfg. vol. 54, no. 5, (Nov. 1954), pp. 136-143.

5. Gordon, B. M.: "Adapting Digital Techniques for Automatic Controls - II", Electrical Mfg. vol. 54, no. 6 (Dec. 1954), pp. 120-125.

6. Amble, O.: "On a Principle of Connexion for Brush Integrators", J. Sci. Instruments vol. 23 (Dec. 1946), pp. 284-287.

7. Michel, J. G. H.: "Extensions on Differential Analyzer Technique", J. Sci. Instruments vol. 25 (Oct. 1948), pp. 357-361.

8. Mergler, H. W.: "A Differential Analyzer", M. S. Thesis, Case Inst. of Technology, 1949.

9. Arnstein, W., Mergler, H. W., and Singer, B.: "Ditital Linear Interpolation and the Binary Rate Multiplier", Control Engineering vol. II, no. 6 (June 1964), pp. 79-83.

10. Mergler, H. W.: Digital Control Systems Engineering, Volumes I and II, Case Inst. of Tech., Cleveland, Ohio 1961.

11. Mergler, H. W.: "A Digital-Analog Machine Tool Control System", Proc. of Western Joint Computer Conf., A.I.E.E., 1954, pp. 46-49.

12. Yang Hsi-Zeng: "A Digital Computer for Programming Second-Degree Curves", Automatika i Telemekhania, vol. 22, no. 3 (March 1961), pp. 309-317.

13. Ragazzini, J. R., and Franklin, G. F.: Sampled-Data Control Systems. New York: McGraw-Hill Book Co., Inc., 1958.

14. Whitaker, E. and Robinson, G.: Calculus of Observations. New York: D. Van Nostrand, Inc., 1944.

15. Henrici, P.: Discrete Variable Methods in Ordinary Differential Equations. New York: John Wiley & Sons, Inc., 1962.

16. Nelson, D. J.: "DDA Error Analysis Using Sampled Data Techniques", Proc. Spring Joint Computer Conference, AFIPS, 1962, pp. 365-373.

17. Brown, E. L.: Digital Computer Design. New York: Academic Press, 1963.

18. Moshos, G. J.: "Analog Interpolator for Automatic Control", JACM, vol. 2, no. 2 (April 1955).

19. Huskey, H. D., and Korn, G. A.: Computer Handbook. New York: McGraw Hill Book Co., Inc., 1962.

APPENDIX A

MULTIPLICATION ERROR BOUNDS (ZERO STARTING)

In this section the error equation of an n-stage BRM whose counter starts with zero will be analyzed with the objective of obtaining tight error bounds. Nevertheless, some of the intermediate results which will be obtained in this section are interesting in their own right. Because this analysis is involved, we will proceed formally. The basic outline is to use Eq. (2.8) to find the points where the maximum positive value is attained and then evaluate the equation at these points. In a similar manner, Eq. (2.10) will be used to find the minimum negative value.

We begin by stating and proving Lemma A.1.

Lemma A.1 A sufficient condition for E given by Eq. (2.8) to attain its maximum value is that $x_i = y_{-i}$.

Eq. (2.8) may be rewritten as a bilinear expression such that the terms which are dependent on either x_i or y_{-i} are grouped together. The quanity A in the resultant expression is independent of either x_i or y_{-i}.

$$E(x_i, y_{-i}) = A + \frac{1}{2} x_i y_{-i} - \frac{1}{2} x_i \left(\frac{1}{2} y_{-i-1} + \frac{1}{4} y_{-i-2} + \cdots + \frac{1}{2^{i-n}} y_{-n} \right) - \frac{1}{2} y_{-i} \left(\frac{1}{2} x_{i-1} + \frac{1}{4} x_{i-2} + \cdots + \frac{1}{2^{i-1}} x_1 \right) \tag{A.1}$$

By direct evaluation the value of this expression is as follows:

$$E(0,0) = A$$

$$E(0,1) = A - \frac{1}{2}\left(\frac{1}{2}\,y_{-i-1} + \cdot\cdot\cdot + \frac{1}{2^{i-n}}\,y_{-n}\right)$$

$$E(1,0) = A - \frac{1}{2}\left(\frac{1}{2}\,x_{i-1} + \cdot\cdot\cdot + \frac{1}{2^{i-1}}\,x_1\right)$$

Moreover, for the specific case when i is n then the value of Eq. (A.1) is

$$E(0,0) = A$$

$$E(0,1) = A$$

$$E(1,0) = A - \frac{1}{2}\left(\frac{1}{2}\,x_{n-1} + \cdot\cdot\cdot + \frac{1}{2^{n-1}}\,x_1\right)$$

$$E(1,1) = A + \frac{1}{2} - \frac{1}{2}\left(\frac{1}{2}\,x_{n-1} + \cdot\cdot\cdot + \frac{1}{2^{n-1}}\,x_1\right)$$

This lemma is proved by observing that the value of E when $x_i = y_{-i} = 0$ is always greater than or equal to the value of E in both cases when $x_i \neq y_{-i}$ and that for the nth component the value of E is always greater when $x_n = y_{-n} = 1$.

Based on this lemma, the maximum value of Eq. (2.8) will be found by finding the maximum of the quadratic form expression

$$Q(x_1,x_2,\ .\ .\ .,x_n) = (x_1,x_2,\ .\ .\ .,x_n)M\begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix} \tag{A.2}$$

<u>Theorem A.1</u> For all values of the components x_i,

$$Q(1,x_2,x_3,\ .\ .\ .,x_n) > Q(0,x_2,x_3,\ .\ .\ .,x_n)$$

Eq. (A.2) may be rewritten such that A is a quadratic expression independent of x_1.

$$Q(x_1,x_2, \ldots, x_n) = A + x_1\left(\frac{1}{2} - \frac{1}{4}x_2 - \frac{1}{8}x_3 \ldots - \frac{1}{2^n}x_n\right) \tag{A.3}$$

By direction evaluation of Eq. (A.3)

$$Q(0,x_2, \ldots, x_n) = A$$

$$Q(1,x_2, \ldots, x_n) = A + \left(\frac{1}{2} - \frac{1}{4}x_2 \ldots - \frac{1}{2^n}x_n\right)$$

Since

$$\frac{1}{2} - \frac{1}{4}x_2 - \frac{1}{8}x_3 - \ldots - \frac{1}{2^n}x_n > 0$$

then

$$Q(1,x_1,x_2, \ldots, x_n) > Q(0,x_1,x_2, \ldots, x_n)$$

<u>Theorem A.2</u> For all values of the components x_i,

$$Q(x_1,x_2, \ldots, x_{n-1},1) > Q(x_1,x_2, \ldots, x_{n-1},0)$$

This proof proceeds similar to Theorem A.1. Q may be rewritten as

$$Q(x_1,x_2, \ldots, x_n) = A + x_n\left(\frac{1}{2} - \frac{1}{4}x_{n-1} - \ldots - \frac{1}{2^n}x_1\right) \tag{A.4}$$

where A is independent of x_n.

By direct evaluation

$$Q(x_1,x_2, \ldots, x_{n-1},0) = A$$

$$Q(x_1,x_2, \ldots, x_{n-1},1) = A + \frac{1}{2} - \frac{1}{4}x_{n-1} - \ldots - \frac{1}{2^n}x_1$$

Therefore

$$Q(x_1,x_2, \ldots, x_{n-1},1) > Q(x_1,x_2, \ldots, x_{n-1},0)$$

<u>Theorem A.3</u> For all values of the components x_i,

$$Q(1,x_2,x_3, \ldots, x_{n-1},1) = Q(1,\bar{x}_2,\bar{x}_3, \ldots, \bar{x}_{n-1},1)$$

where $\bar{x}_i$ is the complement of x_i.

The difference

$$Q(1,x_2,x_3, \ldots,x_{n-1},1) - Q(1,\bar{x}_2,\bar{x}_3, \ldots,\bar{x}_{n-1},1) =$$

$$(1,x_2, \ldots,x_{n-1},1)M \begin{pmatrix} 1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-1} \\ 1 \end{pmatrix} - (1,\bar{x}_2, \ldots,\bar{x}_{n-1},1)M \begin{pmatrix} 1 \\ \bar{x}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{n-1} \\ 1 \end{pmatrix} \tag{A.5}$$

may be written equivalently as

$$(x_2, \ldots,x_{n-1})K \begin{pmatrix} x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-1} \end{pmatrix} - (\bar{x}_2, \ldots,\bar{x}_{n-1})K \begin{pmatrix} \bar{x}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_{n-1} \end{pmatrix} \tag{A.6}$$

where

$$K = \begin{pmatrix} \left(\frac{1}{2}-\frac{1}{2^2}-\frac{1}{2^{n-1}}\right) & -\frac{1}{4} & -\frac{1}{8} & & \cdots & \cdot & \cdots & \cdot \\ 0 & \left(\frac{1}{2}-\frac{1}{2^3}-\frac{1}{2^{n-2}}\right) & -\frac{1}{4} & & \cdots & \cdot & \cdots & \cdot \\ \cdot & \cdot & \left(\frac{1}{2}-\frac{1}{2^4}-\frac{1}{2^{n-3}}\right) & -\frac{1}{4} & \cdots & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdot & \ddots & & & & \\ & & & & & \left(\frac{1}{2}-\frac{1}{2^i}-\frac{1}{2^{n-i+1}}\right) & \cdots & \vdots \\ & & & & & & \ddots & \\ 0 & \cdots & & & & & 0 & \left(\frac{1}{2}-\frac{1}{2^{n-1}}-\frac{1}{2^2}\right) \end{pmatrix}$$

Before proceeding the identity

$$x_i x_j - \bar{x}_i \bar{x}_j = x_i - \bar{x}_j \tag{A.7}$$

is verified by direct computation.

$x_i x_j$	$x_i x_j - \bar{x}_i \bar{x}_j$	$x_i - \bar{x}_j$
0 0	-1	-1
0 1	0	0
1 0	0	0
1 1	1	1

Expanding Eq. (A.5) and using Eq. (A.7) to simplify the cross product terms, then a typical term x_i may be written as

$$x_i\left(\frac{1}{2} - \frac{1}{2^i} - \frac{1}{2^{n-i+1}} - \frac{1}{4} - \frac{1}{8} - \cdot\cdot\cdot\cdot - \frac{1}{2^{n-i}}\right)$$

$$- \bar{x}_i\left(\frac{1}{2} - \frac{1}{2^i} - \frac{1}{2^{n-i+1}} - \frac{1}{4} - \frac{1}{8} - \cdot\cdot\cdot\cdot - \frac{1}{2^{i-1}}\right)$$

$$= x_i\left\{-\frac{1}{2^i} + \frac{1}{2^{n-i+1}}\right\} + \bar{x}_i\left\{\frac{1}{2^{n-i+1}} - \frac{1}{2^i}\right\} = (x_i + \bar{x}_i)\left\{\frac{1}{2^{n-i+1}} - \frac{1}{2^i}\right\} \tag{A.8}$$

Since $(x_i + \bar{x}_i) = 1$, then Eq. (A.6) is independent of the x_i variable and the contribution from this term is

$$\left\{\frac{1}{2^{n-i+1}} - \frac{1}{2^i}\right\} \tag{A.9}$$

Similarly the contribution to the difference expressed in Eq. (A.6) from the term involving x_{n-i+1} is

$$\left\{\frac{1}{2^i} - \frac{1}{2^{n-i+1}}\right\} \tag{A.10}$$

Consequently, the contribution to the difference expressed by Eq. (A.6) by each element may be paired by the contribution from another element such as to cancel each other out of the expression. If n is odd, then the middle term cannot be paired. But since this is the $(n + 1)/2$ term, then by Eq. (A.9) its contribution is

$$\left\{\frac{1}{2^{(n+1)/2}} - \frac{1}{2^{(n+1)/2}}\right\} = 0$$

Therefore, the value of the difference shown by Eq. (A.5) is equal to zero. This implies that

$$Q(1,x_2, \ . \ . \ .,x_{n-1},1) = Q(1,\bar{x}_2, \ . \ . \ .,\bar{x}_{n-1},1)$$

<u>Lemma A.2.</u> For $v = x_2, x_3, \ldots, x_i$ and $a = 1,0,1,0, \ldots, 1,0$

$$Q(1,\bar{v},0,a,1) \geq Q(1,v,1,a,1)$$

where $\bar{v}$ and $\bar{a}$ are the component by component complement of v and a, respectively.

By Theorem A.3 it is noted that

$$Q(1,v,1,a,1) = Q(1,\bar{v},0,\bar{a},1)$$

Therefore, the difference between the two quadratic forms of the lemma can be expressed as

$$\delta = Q(1,\bar{v},0,a,1) - Q(1,\bar{v},0,\bar{a},1) \tag{A.11}$$

Partitioning the M matrix of Eq. (A.11) such that M_1 and M_2 are compatible with the vectors, then δ may be written as

$$\delta = (1,\bar{v},0,a,1)M_1\begin{pmatrix}1\\ \bar{v}\\ 0\end{pmatrix} + (1,\bar{v},0,a,1)M_2\begin{pmatrix}a\\ 1\end{pmatrix}$$

$$- (1,\bar{v},0,\bar{a},1)M_1\begin{pmatrix}1\\ \bar{v}\\ 0\end{pmatrix} - (1,\bar{v},0,\bar{a},1)M_2\begin{pmatrix}\bar{a}\\ 1\end{pmatrix} \tag{A.12}$$

But it will be noted that

$$(1,\bar{v},0,a,1)M_1\begin{pmatrix}1\\ \bar{v}\\ 0\end{pmatrix} = (1,\bar{v},0,\bar{a},1)M_1\begin{pmatrix}1\\ \bar{v}\\ 0\end{pmatrix}$$

Therefore

$$\delta = (1,\bar{v},0,a,1)M_2\begin{pmatrix}a\\ 1\end{pmatrix} - (1,\bar{v},0,\bar{a},1)M_2\begin{pmatrix}\bar{a}\\ 1\end{pmatrix} \tag{A.13}$$

It will now be proved by induction on the length of a that $\delta \geq 0$. It may be immediately verified that $\delta = 0$ for a of length zero. Assume that $\delta_k \geq 0$ where δ_k is the value of δ when a is of length $2k$. It will now be verified that $\delta_{k+1} \geq 0$.

$$\delta_{k+1} = \delta_k + \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \vdots \\ \bar{x}_i \\ 0_{i+1} \\ 1_{i+2} \\ 0_{i+3} \\ \vdots \\ 0_{i+2k-1} \\ 1_{i+2k} \\ 0_{i+2k+1} \\ 1_{i+2k+2} \\ 0_{i+2k+3} \\ 1_{i+2k+4} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+4} \\ -1/2^{i+2k+3} \\ -1/2^{i+2k+2} \\ \vdots \\ -1/2^{2k+5} \\ -1/2^{2k+4} \\ -1/2^{2k+3} \\ -1/2^{2k+2} \\ \vdots \\ -1/64 \\ -1/32 \\ -1/16 \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} - \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \vdots \\ \bar{x}_i \\ 0_{i+1} \\ 0_{i+2} \\ 1_{i+3} \\ \vdots \\ 1_{i+2k-1} \\ 0_{i+2k} \\ 1_{i+2k+1} \\ 0_{i+2k+2} \\ 1_{i+2k+3} \\ 1_{i+2k+4} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+3} \\ \vdots \\ \\ \vdots \\ \\ \vdots \\ -1/4 \\ 1/2 \\ 0 \end{pmatrix}$$

$$- \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \vdots \\ \bar{x}_i \\ 0_{i+1} \\ 0_{i+2} \\ 1_{i+3} \\ \vdots \\ 1_{i+2k-1} \\ 0_{i+2k} \\ 1_{i+2k+1} \\ 0_{i+2k+2} \\ 1_{i+2k+3} \\ 1_{i+2k+4} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+4} \\ \vdots \\ \\ \vdots \\ \\ \vdots \\ -1/4 \\ 1/2 \end{pmatrix} + \begin{pmatrix} 1 \\ x_2 \\ \vdots \\ x_i \\ 0_{i+1} \\ 0_{i+2} \\ 1_{i+3} \\ \vdots \\ 0_{i+2k} \\ 1_{i+2k+1} \\ 1_{i+2k+2} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+2} \\ -1/2^{i+2k+1} \\ \vdots \\ -1/2^{2k+3} \\ -1/2^{2k+2} \\ -1/2^{2k+1} \\ -1/2^{2k} \\ \vdots \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} \qquad (A.14)$$

The 0's and 1's in Eq. (A.14) are subscripted to show their position in the vector, and the vector itself is displayed as a column rather than a row in order to show the correspondence between the terms which must be multiplied to form the value.

Multiplying out the terms of Eq. (A.14), then

$$\begin{aligned}\delta_{k+1} = \delta_k + \frac{1}{2} - \frac{1}{8}\left(1 + \frac{1}{4} + \cdots + \frac{1}{2^{2k}}\right) - \frac{1}{2^{i+2k+4}} \\ - \frac{1}{2^{2k+5}}\left(\bar{x}_i + \frac{1}{2}\bar{x}_{i-1} + \cdots + \frac{1}{2^{i-2}}\bar{x}_2\right) \\ + \frac{1}{2} - \frac{1}{4}\left(1 + \frac{1}{4} + \cdots + \frac{1}{2^{2k-2}}\right) - \frac{1}{2^{i+2k+2}} \\ - \frac{1}{2^{2k+5}}\left(\bar{x}_i + \frac{1}{2}\bar{x}_{i-1} + \cdots + \frac{1}{2^{i-2}}\bar{x}_2\right) \\ - \frac{3}{4} + \frac{3}{16}\left(1 + \frac{1}{4} + \cdots + \frac{1}{2^{2k-2}}\right) - \frac{3}{2^{i+2k+4}} \\ - \frac{3}{2^{2k+5}}\left(\bar{x}_i + \frac{1}{2}\bar{x}_{i-1} + \cdots + \frac{1}{2^{i-2}}\bar{x}_2\right)\end{aligned} \tag{A.15}$$

Using the relations

$$1 + \frac{1}{4} + \cdots + \frac{1}{2^{2k}} = \frac{4}{3}\left(1 - \frac{1}{2^{2k+2}}\right)$$

and

$$1 + \frac{1}{4} + \cdots + \frac{1}{2^{2k-2}} = \frac{4}{3}\left(1 - \frac{1}{2^{2k}}\right)$$

then Eq. (A.15) becomes

$$\delta_{k+1} = \delta_k + \frac{1}{2^{2k+3}} - \frac{1}{2^{i+2k+3}} - \frac{1}{2^{2k+4}}\sum_{j=0}^{i-2}\frac{1}{2^j}\bar{x}_{i-j}$$

But by direct evaluation

$$\frac{1}{2^{2k+4}} \sum_{j=0}^{i-2} \frac{1}{2^j} \bar{x}_{i-j} + \frac{1}{2^{2k+4}} \frac{1}{2^{i-1}} < \frac{2}{2^{2k+4}} = \frac{1}{2^{2k+3}}$$

Therefore $\delta_{k+1} > 0$.

Lemma A.3. For $v = x_2, x_3, \ldots, x_i$ and $a = 1,0,1,0, \ldots, 1,0$

$$Q(1,\bar{v},1,0,a,1) \geq Q(1,v,0,0,a,1)$$

where $\bar{v}$ and $\bar{a}$ are defined as in Lemma A.2.

Proceeding in a manner similar to Lemma A.2, it is first noted by Theorem A.3 that

$$Q(1,v,0,0,a,1) = Q(1,\bar{v},1,1,\bar{a},1)$$

Therefore, the difference between the two quadratic forms of the lemma can be expressed as

$$\delta = Q(1,\bar{v},1,0,a,1) - Q(1,\bar{v},1,1,\bar{a},1) \tag{A.16}$$

Partitioning the M matrix of Eq. (A.16) such that M_1 and M_2 are compatible with the vectors, then δ may be written as

$$\delta = (1,\bar{v},1,0,a,1)M_1\begin{pmatrix}1\\ \bar{v}\\ 1\end{pmatrix} + (1,\bar{v},1,0,a,1)M_2\begin{pmatrix}0\\ a\\ 1\end{pmatrix}$$

$$- (1,\bar{v},1,1,\bar{a},1)M_1\begin{pmatrix}1\\ \bar{v}\\ 1\end{pmatrix} - (1,\bar{v},1,1,\bar{a},1)M_2\begin{pmatrix}1\\ \bar{a}\\ 1\end{pmatrix} \tag{A.17}$$

But it will be noted that

$$(1,\bar{v},1,0,\bar{a},1)M_1\begin{pmatrix}1\\ \bar{v}\\ 1\end{pmatrix} = (1,\bar{v},1,1,\bar{a},1)M_1\begin{pmatrix}1\\ \bar{v}\\ 1\end{pmatrix}$$

Therefore

$$\delta = (1,\bar{v},1,0,a,1)M_2\begin{pmatrix}0\\ a\\ 1\end{pmatrix} - (1,\bar{v},1,1,\bar{a},1)M_2\begin{pmatrix}1\\ \bar{a}\\ 1\end{pmatrix} \tag{A.18}$$

Using δ_k to denote the value of δ when a is of length 2k, it will now be proved by induction on the length of a that

$$\delta_k \geq \frac{1}{2^{2k+3}} \bar{x}_i + \frac{1}{2^{2k+4}} \bar{x}_{i-1} + \ . \ . \ . + \frac{1}{2^{i+2k+1}} \bar{x}_2 + \frac{1}{2^{i+2k+2}} > 0$$

(A.19)

First we note that for a of length zero

$$\delta_0 = \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 0_{i+2} \\ 1_{i+3} \end{pmatrix} \begin{pmatrix} -1/2^{i+3} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ -1/16 \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} - \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 1_{i+2} \\ 1_{i+3} \end{pmatrix} \begin{pmatrix} -1/2^{i+2} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ -1/8 \\ -1/4 \\ 1/2 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 1_{i+2} \\ 1_{i+3} \end{pmatrix} \begin{pmatrix} -1/2^{i+3} \\ -1/2^{i+2} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ -1/16 \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix}$$

$$= \frac{1}{2^3} \bar{x}_i + \frac{1}{2^4} \bar{x}_{i-1} + \ . \ . \ . + \frac{1}{2^{i+1}} \bar{x}_2 + \frac{1}{2^{i+2}} \qquad \text{(A.20)}$$

The notation used in Eq. (A.20) is similar to that used in proving Lemma A.2.

Assume that $\delta_k > 0$, it will now be shown that $\delta_{k+1} > 0$

$$
\delta_{k+1} = \delta_{k} + \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 0_{i+2} \\ 1_{i+3} \\ 0_{i+4} \\ \cdot \\ \cdot \\ \cdot \\ 1_{i+2k+1} \\ 0_{i+2k+2} \\ 1_{i+2k+3} \\ 0_{i+2k+4} \\ 1_{i+2k+5} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+5} \\ -1/2^{i+2k+4} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ -1/2^{2k+6} \\ \cdot \\ \cdot \\ \\ \cdot \\ \\ \\ \\ \cdot \\ \cdot \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} - \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 1 \\ 0 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 1 \\ 0 \\ 1 \\ 1_{i+2k+5} \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+4} \\ \cdot \\ \cdot \\ \cdot \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \cdot \\ \cdot \\ \cdot \\ -1/4 \\ 1/2 \\ 0 \end{pmatrix}
$$

$$
- \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1 \\ 1 \\ 0 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} -1/2^{i+2k+5} \\ \\ \\ \\ \\ \\ \\ \\ \\ \cdot \\ \\ \cdot \\ \cdot \\ \\ \cdot \\ \cdot \\ \cdot \\ -1/4 \\ 1/2 \end{pmatrix} + \begin{pmatrix} 1 \\ \bar{x}_2 \\ \bar{x}_3 \\ \cdot \\ \cdot \\ \cdot \\ \bar{x}_i \\ 1_{i+1} \\ 1_{i+2} \\ 0_{i+3} \\ 1_{i+4} \\ \cdot \\ \cdot \\ \cdot \\ 0_{i+2k+1} \\ 1_{i+2k+2} \\ 1_{i+2k+3} \end{pmatrix} \begin{pmatrix} 1/2^{i+2k+3} \\ \cdot \\ \cdot \\ \\ \\ \\ \\ \\ \\ \\ \cdot \\ \\ \cdot \\ \cdot \\ \cdot \\ -1/4 \\ 1/2 \end{pmatrix} \qquad \text{(A.21)}
$$

Multiplying out Eq. (A.21), then δ_{k+1} becomes

$$\delta_{k+1} = \delta_k + \frac{1}{2} - \frac{1}{8}\left(1 + \frac{1}{2^2} + \ldots + \frac{1}{2^{2k+2}}\right) - \frac{1}{2^{2k+6}}\,\bar{x}_i - \ldots$$

$$- \frac{1}{2^{i+2k+4}}\,\bar{x}_2 - \frac{1}{2^{i+2k+5}} - \frac{3}{4} + \frac{3}{16}\left(1 + \frac{1}{2^2} + \ldots + \frac{1}{2^{2k}}\right) - \frac{3}{2^{2k+5}}$$

$$+ \frac{3}{2^{2k+6}}\,\bar{x}_i + \ldots + \frac{3}{2^{i+2k+4}}\,\bar{x}_2 + \frac{3}{2^{i+2k+5}} + \frac{1}{2}$$

$$- \frac{1}{4}\left(1 + \frac{1}{2^2} + \ldots + \frac{1}{2^{2k}}\right) - \frac{1}{2^{2k+3}} - \frac{4}{2^{2k+6}}\,\bar{x}_i - \ldots$$

$$- \frac{4}{2^{i+2k+4}}\,\bar{x}_2 - \frac{4}{2^{i+2k+5}} \qquad \text{(A.22)}$$

Equation (A.22) may be reduced to

$$\delta_{k+1} = \delta_k - \frac{1}{2^{2k+5}}\,\bar{x}_i - \ldots - \frac{1}{2^{i+2k+3}}\,\bar{x}_2 - \frac{1}{2^{i+2k+4}} \qquad \text{(A.23)}$$

Using Eq. (A.19), then the right hand side of Eq. (A.23) becomes

$$> \frac{1}{2^{2k+3}}\,\bar{x}_i + \ldots + \frac{1}{2^{i+2k+1}}\,\bar{x}_2 + \frac{1}{2^{i+2k+2}} - \frac{1}{2^{2k+5}}\,\bar{x}_i - \ldots$$

$$- \frac{1}{2^{i+2k+3}}\,\bar{x}_2 - \frac{1}{2^{i+2k+4}} > \frac{1}{2^{2k+5}}\,\bar{x}_i + \ldots \quad \frac{1}{2^{i+2k+3}}\,\bar{x}_2 + \frac{1}{2^{i+2k+4}} > 0$$

<u>Theorem A.4.</u> There exists a v^* such that for all v

$$Q(1,v^*,0,a,1) \geq Q(1,v,x_{i+1},a,1)$$

where v and a are defined as before.

First, we note that $Q(1,v^{**},0,a,1) \geq Q(1,v,1,a,1)$ because suppose it were false then there would exist a $\overset{*}{v^{**}}$ such that for all of v

$$Q(1,\overset{*}{v^{**}},1,a,1) > Q(1,v,0,a,1)$$

But from Lemma A.2

$$Q(1,\overset{*}{\bar{v}^{**}},0,a,1) \geq Q(1,\overset{*}{v^{**}},1,a,1)$$

Therefore, a contradiction exists.

Moreover, we note that

$$Q(1,\overset{**}{v^{**}},0,a,1) \geq Q(1,v,0,a,1)$$

that is, there is a largest.

Therefore, Theorem A.4 is proved by choosing either v^{**} or $\overset{**}{v^{**}}$ for v^*; that is, whichever make $Q(1,v^*,0,a,1)$ the largest.

<u>Theorem A.5.</u> There exists a v^* such that for all v

$$Q(1,v^*,1,0,a,1) \geq Q(1,v,x_{i+1},0,a,1)$$

First, we note that $Q(1,v^{**},1,0,a,1) \geq Q(1,v,0,0,a,1)$ because suppose it were false then there would exist a $\overset{*}{v^{**}}$ such that for all v

$$Q(1,\overset{*}{v^{**}},0,0,a,1) > Q(1,v,1,0,a,1)$$

But from Lemma A.3

$$Q(1,\overset{*}{\bar{v}^{**}},1,0,a,1) \geq Q(1,\overset{*}{v^{**}},0,0,a,1)$$

Therefore, a contradiction exist.

Moreover, a $\overset{**}{v^{**}}$ can be choosen such that

$$Q(1,\overset{**}{v^{**}},0,0,a,1) \geq Q(1,v,0,0,a,1)$$

Therefore

$$Q(1,v^*,1,0,a,1) \geq Q(1,v,x_{i+1},0,a,1)$$

<u>Theorem A.6.</u> There exists a v^* such that for all of v

$$Q(1,v^*,0,1) \geq Q(1,v,x_{i+1},1)$$

First, we note that there exist a v^{**} such that $Q(1,v^{**},0,1) \geq Q(1,v,1,1)$, because suppose it were false then there would exist a v^{***} such that for all v

$$Q(1,v^{***},1,1) > Q(1,v,0,1)$$

But by Theorem A.3

$$Q(1,\bar{v}^{**},0,1) = Q(1,v^{**},1,1)$$

Therefore, a contradiction exists.

Moreover, a v^{****} can be chosen such that

$$Q(1,v^{****},0,1) \geq Q(1,v,0,1)$$

Therefore

$$Q(1,v^{*},0,1) \geq Q(1,v,x_{i+1},1)$$

It will now be demonstrated by an example how these theorems can be used to obtain the value of x such that the error is the maximum positive value. Suppose we consider a 7-stage BRM. By Theorem A.1 and Theorem A.2

$$Q(1,x_2,x_3,x_4,x_5,x_6,1) \geq Q(x_1,x_2,x_3,x_4,x_5,x_6,x_7)$$

by Theorem A.6

$$Q(1,x_2^{*},x_3^{*},x_4^{*},x_5^{*},0,1) \geq Q(1,x_2,x_3,x_4,x_5,x_6,1)$$

by Theorem A.5

$$Q(1,x_2^{**},x_3^{**},x_4^{**},1,0,1) \geq Q(1,x_2^{*},x_3^{*},x_4^{*},x_5^{*},0,1)$$

by Theorem A.4

$$Q(1,x_2^{***},x_3^{***},0,1,0,1) \geq Q(1,x_2^{**},x_3^{**},x_4^{**},1,0,1)$$

by Theorem A.5

$$Q(1,x_2^{****},1,0,1,0,1) \geq Q(1,x_2^{***},x_3^{***},0,1,0,1)$$

by Theorem A.6

$$Q(1,0,1,0,1,0,1) \geq Q(1,x_2^{****},1,0,1,0,1)$$

Putting these inequalities together

$$Q(1,0,1,0,1,0,1) \geq Q(x_1,x_2,x_3,x_4,x_5,x_6,x_7)$$

Moreover, by Theorem A.3

$$Q(1,1,0,1,0,1,1) \geq Q(x_1,x_2,x_3,x_4,x_5,x_6,x_7)$$

Using the theorems in the pattern illustrated by the example, it is easy to verify that the maximum positive value will occur at the points shown in Table 2.4.

The maximum positive value of the error may be expressed concisely as follows:

Let $E_{max}(k)$ denote this value for an k stage BRM. If k is odd, then

$$E_{max}(k+2) = E_{max}(k) + \begin{vmatrix} 1_1 \\ 0_2 \\ 1_3 \\ \cdot \\ \cdot \\ \cdot \\ 1_k \\ 0_{k+1} \\ 1_{k+2} \end{vmatrix} \begin{pmatrix} -1/2^{k+2} \\ \\ \\ \cdot \\ \cdot \\ \cdot \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} \tag{A.24}$$

Evaluating Eq. (A.25), this yields the difference equation

$$E_{max}(k+2) = E_{max}(k) + \frac{1}{3}\left(1 + \frac{1}{2^{k+2}}\right) \tag{A.25}$$

Solving Eq. (A.30) for a n stage BRM in terms of $E_{max}(1)$ yields:

$$E_{max}(n) = E_{max}(1) - \frac{1}{9} + \frac{n}{6} - \frac{1}{9 \cdot 2^n}$$

But $E_{max}(1) = 1/2$

Therefore, the maximum positive error of an n-stage BRM where n is odd is:

$$E_{max}(n) = \frac{7}{18} + \frac{n}{6} - \frac{1}{9 \cdot 2^n} \qquad \text{(A.26)}$$

If k is even, then

$$E_{max}(k+2) = E_{max}(k) + \begin{pmatrix} 1_1 \\ 1_2 \\ 0_3 \\ 1_4 \\ \cdot \\ \cdot \\ \cdot \\ 0_{k-1} \\ 1_k \\ 0_{k+1} \\ 1_{k+2} \end{pmatrix} \begin{pmatrix} -1/2^{k+2} \\ -1/2^{k+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ -1/8 \\ -1/4 \\ 1/2 \end{pmatrix} = E_{max}(k) + \frac{1}{3} - \frac{1}{3}\,\frac{1}{2^{k+2}}$$

(A.27)

Solving this difference equation for an n-stage BRM in terms of $E_{max}(2)$ and then evaluating the resultant expression for $E_{max}(2) = 3/4$, yields

$$E_{max}(n) = \frac{7}{18} + \frac{n}{6} + \frac{1}{9 \cdot 2^n} \qquad \text{(A.28)}$$

Combining equations (A.26) and (A.28) gives a closed form equation for the maximum positive error of a n-stage BRM.

$$E_{max}(n) = \frac{7}{18} + \frac{n}{6} + \frac{(-1)^n}{9 \cdot 2^n} \qquad \text{(A.29)}$$

By applying Eq. (2.10) the minimum negative value for a n stage BRM can be obtained. Comparing the form of Eq. (2.8) to Eq. (2.10), it is seen that our previous results can be utilized with a slight modification. In particular, the value of the minimum is equal to the negative of the maximum and occur at points which are the 2's complement of the maximum value. Consequently, the minimum negative values will occur at the points shown in Table 2.5.

APPENDIX B

MULTIPLICATION ERROR BOUNDS (ARBITRARY STARTING)

The error formulas given by Eqs.(2.12) and (2.13) expresses the multiplication error of a BRM whose counter starts with an arbitrary value. Eq. (2.12) is the error formula resulting when the maximum value of the actual output is considered at the points of discontinuities. Eq. (2.13) is the companion equation resulting when the minimum value of the actual output is considered at these points. In this section these error formulas will be analyzed with the objective of obtaining error bounds for a BRM with this added degree of freedom. We begin by analyzing Eq. (2.12) to obtain the maximum positive error of an n stage BRM.

It is convenient for this discussion to define a vector b such that

$$\begin{pmatrix} b_{-1} \\ b_{-2} \\ \cdot \\ \cdot \\ \cdot \\ b_{-n} \end{pmatrix} = M \begin{pmatrix} y_{-1} \\ y_{-2} \\ \cdot \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \qquad \text{(B.1)}$$

<u>Theorem B.1</u> For all x_k and x_{Sk} in Eq. (2.12), $G \leq \sum_{i=1}^{n} |b_{-i}|$.

Moreover, if $y_{-k} = x_k = \bar{x}_{Sk}$ then $G = \sum_{i=1}^{n} |b_{-i}|$ where $\bar{x}_{Sk}$ denotes the complement of x_{Sk}.

The elements of the vector defined by Eq. (B.1) are

$$b_{-1} = \frac{1}{2} y_{-1} - \left(\frac{1}{4} y_{-2} + \frac{1}{8} y_{-3} + \cdots + \frac{1}{2^n} y_{-n}\right)$$

$$b_{-2} = \frac{1}{2} y_{-2} - \left(\frac{1}{4} y_{-3} + \frac{1}{8} y_{-4} + \cdots + \frac{1}{2^{n-1}} y_{-n}\right)$$

$$\vdots$$

$$b_{-k} = \frac{1}{2} y_{-k} - \left(\frac{1}{4} y_{-k-1} + \frac{1}{8} y_{-k-2} + \cdots + \frac{1}{2^{n-k+1}} y_{-n}\right)$$

$$\vdots$$

$$b_{-n} = \frac{1}{2} y_{-n}$$

Since $1/2 > 1/4 + \ldots + 1/2^n$ then

$$\begin{aligned} b_{-k} &\leq 0 \quad \text{if} \quad y_{-k} = 0 \\ &> 0 \quad \text{if} \quad y_{-k} = 1 \end{aligned} \tag{B.2}$$

It is next noted that each element $(x_k - x_{Sk})$ may have only three possible values; that is, 0,1, or -1. This may be verified by direct computation.

x_k	x_{Sk}	$x_k - x_{Sk}$
0	0	0
0	1	-1
1	0	1
1	1	0

(B.3)

Since $|x_k - x_{Sk}| \leq 1$ then

$$G = (x_1 - x_{S1})b_{-1} + (x_2 - x_{S2})b_{-2} + \cdots + (x_n - x_{Sn})b_{-n} \leq \sum_{i=1}^{n} |b_{-i}| \tag{B.4}$$

A sufficient condition for G to attain the upper bound of Eq. (B.4), that is, $\sum_{i=1}^{n} |b_{-i}|$, is that:

$$(x_k - x_{Sk}) = -1 \quad \text{if} \quad b_{-k} \leq 0$$
$$= +1 \quad \text{if} \quad b_{-k} > 0 \qquad (B.5)$$

Combining (B.2), (B.3), and (B.5) results in

y_{-k}	b_{-k}	$x_k - x_{Sk}$	x_k	x_{Sk}
0	≤ 0	-1	0	1
1	≥ 0	+1	1	0

Therefore, $y_{-k} = x_k = \bar{x}_{Sk}$ is a sufficient condition for $G = \sum_{i=1}^{n} |b_{-i}|$.

As a consequence of Theorem B.1 the maximum of G; denoted by G_{max}, is such that $G_{max} = \max_{y} \sum_{i=1}^{n} |b_{-i}|$. The procedure which is to be followed is to find the value y where the maximum of $\sum_{i=1}^{n} |b_{-i}|$ is attained and then evaluating this function. In order to aid this analysis the notation $b_{-i}(y)$ is introduced, where $y = y_{-1}, y_{-2}, \ldots, y_{-n}$ and $b_{-i}(y)$ denotes the value b_{-i} for the vector $(y_{-1}, y_{-2}, \ldots, y_{-n})$. For an n stage BRM, all possible $b_{-i}(y)$ values may be obtained by multiplying M with all possible values of y. We will call this particular matrix B_n.

$$B_n = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} & -\frac{1}{8} & \cdots & -\frac{1}{2^n} \\ 0 & \frac{1}{2} & -\frac{1}{4} & & -\frac{1}{2^{n-1}} \\ \vdots & & \ddots & & \\ & & & \frac{1}{2} & -\frac{1}{4} \\ 0 & \cdots & & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 & 0 & \cdots & 1 & 1 & \cdots & 1 \\ 0 & 0 & & 0 & 0 & & 1 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 1 & \cdots & 0 & 0 & \cdots & 1 \\ 1 & 0 & \cdots & 0 & 1 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} b_{-1}(1) & b_{-1}(2) & \cdots & b_{-1}(2^n - 1) \\ b_{-2}(1) & b_{-2}(2) & \cdots & b_{-2}(2^n - 1) \\ \vdots & \vdots & & \vdots \\ b_{-n}(1) & b_{-n}(2) & \cdots & b_{-n}(2^n - 1) \end{pmatrix} \tag{B.6}$$

A few examples will help clarify Eq. (B.6)

Two-stage BRM

$$B_2 = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} \\ 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -\frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}$$

Three-stage BRM

$$B_3 = \begin{pmatrix} \frac{1}{2} & -\frac{1}{4} & -\frac{1}{8} \\ 0 & \frac{1}{2} & -\frac{1}{4} \\ 0 & 0 & \frac{1}{2} \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} -\frac{1}{8} & -\frac{1}{4} & -\frac{3}{8} & \frac{1}{2} & \frac{3}{8} & \frac{1}{4} & \frac{1}{8} \\ -\frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & -\frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}$$

Four-stage BRM

$$B_4 = \begin{pmatrix} -\frac{1}{16} & -\frac{2}{16} & -\frac{3}{16} & -\frac{4}{16} & -\frac{5}{16} & -\frac{6}{16} & -\frac{7}{16} & \frac{1}{2} & \frac{7}{16} & \frac{6}{16} & \frac{5}{16} & \frac{4}{16} & \frac{3}{16} & \frac{2}{16} & \frac{1}{16} \\ -\frac{1}{8} & -\frac{1}{4} & -\frac{3}{8} & \frac{1}{2} & \frac{3}{8} & \frac{1}{4} & \frac{1}{8} & 0 & & & & & & & \\ & & & 0 & -\frac{1}{4} & \frac{1}{2} & \frac{1}{4} & 0 & & & & B_3 & & & \\ & B_2 & & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & & & & & & & \end{pmatrix}$$

<u>Theorem B.2</u>. For all values of y, $|b_{-i}(y)| = |b_{-i}(2^n-y)|$

This result follows by induction on the number of stages k. It was shown as an example for the $k = 2$ case. Assume it is true for the $k = n - 1$ case. Then the $k = n$ case is

$$B_n = \begin{pmatrix} -\frac{1}{2^n} & -\frac{2}{2^n} & \cdots & -\frac{2^{n-1}-1}{2^n} & \frac{1}{2} & \frac{2^{n-1}-1}{2^n} & \cdots & \frac{1}{2^n} \\ & & & & 0 & & & \\ & & B_{n-1} & & \vdots & & B_{n-1} & \\ & & & & 0 & & & \end{pmatrix}$$

where this case is partitioned to show its structure. This theorem is obviously true for the first row. The $b_{-i}(y)$ element for the $n - 1$ case is now the $b_{-i-1}(y)$ and the $b_{-i-1}(2^n + y)$ elements of the n case; and the $b_{-i}(2^n - y)$ element of the $n - 1$ case is now the $b_{-i-1}(2^n - y)$ and the $b_{-i-1}(2^n + 2^n - y)$ elements of the B_n case. By the induction hypothesis $|b_{-i}(y)| = |b_{-i}(2^n - y)|$ for the $n - 1$ case. Therefore, these elements for the nth case yield

$$|b_{-i-1}(y)| = |b_{-i-1}(2^{n+1} - y)| \tag{B.7}$$

and

$$|b_{-i-1}(2^n - y)| = |b_{-i-1}(2^n + y)| \tag{B.8}$$

Substituting $u = 2^n - y$ into Eq. (B.8) then

$$|b_{-i-1}(u)| = |b_{-i-1}(2^{n+1} - u)|$$

which completes the proof.

<u>Lemma B.1.</u> There exists a y^* in the domain $0\,1\,0\,.\,.\,.\,0\,0 \le y^* \le 0\,1\,1\,1\,.\,.\,.\,1\,1$ such that

$$\sum_i |b_{-i}(y^*)| \ge \sum_i |b_{-i}(y)|.$$

This theorem states that G_{max} is attained in the domain $0\,1\,0\,.\,.\,.\,0\,0 \le y \le 0\,1\,1\,.\,.\,.\,1\,1.$ As a result of Theorem B.1, the search for a point where G_{max} is attained can be immediately restricted to the y domain $0 < y \le 1\,0\,0\,.\,.\,.\,0\,0.$ Consider B_n for these values of y.

$$B_n = \begin{pmatrix} -\frac{1}{2^n} & -\frac{2}{2^n} & \cdots & -\frac{1}{4} & \cdots & \frac{1}{2} & \cdots \\ & & & \frac{1}{2} & & 0 & \\ & B_L & & 0 & B_R & 0 & \cdots \\ & & & \vdots & & \vdots & \\ & & & 0 & & 0 & \cdots \end{pmatrix}$$

The vector $\begin{pmatrix} -1/4 \\ 1/2 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ results from the y vector $\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$. The vector $\begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ can be immediately ruled out.

The structure $B_L \begin{pmatrix} 1/2 \\ 0 \\ 0 \\ \vdots \end{pmatrix} B_R$ in the above matrix is B_{n-1}.

Because of Theorem B.2, the absolute values of the elements in B_L are identical to the absolute values of the elements B_R. Moreover, since the elements of first row, that is, $|b_{-1}(y)|$, increase as y increases then for each column sum to the left of $\begin{pmatrix} -1/4 \\ 1/2 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$ there is a column sum to the right which exceeds it. Therefore, G_{max} must lie to the right.

As a result of Theorem B.2 and Lemma B.1, there must be at least two values of y where G_{max} is attained. For an n stage BRM, we will call the y value corresponding to G_{max} on the left of $y = 1000 \ldots 00$, L_n; and the one on the right of $y = 100 \ldots 00$, R_n.

Lemma B.2. For an n-stage BRM

$$0R_{n-1} \leq L_n < 0\,1\,1 \ldots 1\,1$$

This result follows from the proof of Lemma B.1. Since the first row $|b_{-1}(y)|$ increasing then G_{max} must lie between the right maximum of B_{n-1} and the rightmost value of Lemma B.1.

Lemma B.3. For an n-stage BRM

$$0\,1\,0\,0 \ldots 0 \leq L_n \leq 0\,1\,L_{n-2}$$

Consider B_n for the values of y of Lemma B.1, that is $0\,1\,0\,0 \ldots 0 \leq y \leq 0\,1\,1 \ldots 1$.

$$B_n = \begin{pmatrix} -\frac{1}{4} & -\frac{2^{n-2}+1}{2^n} & \frac{2^{n-2}+2}{2^n} & - & \cdots & \frac{1}{2} \\ \frac{1}{2} & \frac{2^{n-2}-1}{2^{n-1}} & \frac{2^{n-2}-2}{2^{n-1}} & & \cdots & 0 \\ & & & & & \vdots \\ & & B_{n-2} & & & \\ & & & & & 0 \end{pmatrix}$$

The sums of the absolute value of the first two rows are

$$\frac{3}{4}, \frac{2^{n-1}+2^{n-2}-1}{2^n}, \frac{2^{n-1}+2^{n-2}-2}{2^n}, \ldots$$

Therefore, this is a decreasing sequence. Since by Theorem B.2 and Lemma B.1, B_{n-2} attains at a maximum for at least two values, then G_{max} must lie between the leftmost value of Lemma B.1 and $01L_{n-2}$.

Theorem B.3. For an n-stage BRM

$$OR_{n-1} \leq L_n \leq 01\ L_{n-2}$$

This theorem is the combination of Lemma B.2 and Lemma B.3.

Theorem B.4. R_n and L_n are unique and $OR_{n-1} = L_n = 01\ L_{n-2}$.

This theorem follows immediately from the proofs of Lemma B.2 and Lemma B.3 by using the principle of strong induction as the method of proof; that is, assume it is true for $k < n$ and prove for the case $k = n$.

The values of y where G_{max} is attained can be found by using Theorem B.4. These values are listed in Table B.1.

TABLE B.1 VALUES OF y WHERE G_{max} IS ATTAINED

	$01L_{n-2}$	$0R_{n-1}$	L_n	R_n
n	$y_{-1} \cdots y_{-6}$	$y_{-1} \cdots y_{-6}$	$y_{-1} \cdots y_{-6}$	$y_{-1} \cdots y_{-6}$
1			1	1
2			01	11
3	011	011	011	101
4	0101	0101	0101	1011
5	01011	01011	01011	10101
6	010101	010101	010101	101011

The BRM counter value and starting value corresponding to the y values listed in Table B.1 can be obtained by Theorem B.1. These values are tabulated in Table 2.6.

An equation for G_{max} as a function of n may be obtained by a procedure similar to that used to obtain E_{max}. In particular, using the pattern established above a difference equation may be written for n even, and a difference equation may be written for n odd. Combining the solutions of these difference equations, G_{max} may be obtained as a function of n.

$$G_{max}(n) = \frac{1}{9} + \frac{n}{3} - \frac{(-1)^n}{9 \cdot 2^n} \tag{B.9}$$

Noting the similarity between the rightmost term of Eq. (2.13) to that of Eq. (2.12), one may establish immediately a minimum error bound when the BRM counter starts with an arbitrary value.

$$H_{min}(n) \geq -\frac{10}{9} - \frac{n}{3} + \frac{(-1)^n}{9 \cdot 2^n} \tag{B.10}$$

A tight error bound may be obtained as follows: Expanding Eq. (2.13), then equation for H may be expressed as

$$H = -x_{SR}y_{-R} - \left[(C_1 - C_{S1})b_{-1} + (C_2 - C_{S2})b_{-2} + \cdots + (C_n - C_{Sn})b_{-n}\right] \tag{B.11}$$

x_{SR} identifies the rightmost 1 in the initial value of the BRM counter. It may be simply argued that if a binary number has a rightmost 1 in position R then its 2's complement also has a 1 in that position and, moreover, has zeros at all positions j where $j < R$. Therefore, $x_{SR} = C_{SR}$ and $C_{Sj} = 0$ for all $j < R$.

Eq. (B.11) may be expressed as

$$-H = C_{SR}y_{-R} + \left[C_1b_{-1} + C_2b_{-2} + \ldots + C_{R-1}b_{-R+1}\right] + (C_R - C_{SR})b_{-R} + \left[(C_{R+1} - C_{SR+1})b_{-R-1} + \ldots + (C_n - C_{Sn})b_{-n}\right] \tag{B.12}$$

or alternately as

$$-H = C_1b_{-1} + C_2b_{-2} + \ldots + C_Rb_{-R} + (C_{R+1} - C_{SR+1})b_{-R-1} + \ldots + (C_n - C_{Sn})b_{-n} + C_{SR}\left[\frac{1}{2}y_{-R} + \frac{1}{4}y_{-R-1} + \ldots +\right] \tag{B.13}$$

Therefore

$$\leq |C_1b_{-1}| + |C_2b_{-2}| + \ldots + |C_Rb_{-R}| + |(C_{R+1} - C_{SR+1})b_{-R-1}| + \ldots + |(C_n - C_{Sn})b_{-n}| + C_{SR}\left[\frac{1}{2}y_{-R} + \frac{1}{4}y_{-R-1} + \ldots\right] \tag{B.14}$$

The upper bound of $-H$ is attained when equality is attained in Eq. (B.14). It can be demonstrated by an argument similar to that used in Theorem B.1, that the conditions for equality are

$$y_{-j} = C_j \qquad \text{for all } j$$

and

$$C_{Sj} = \overline{C}_j \qquad \text{for all } j > R$$

Therefore, Eq. (B.14) may be written as

$$-H = (C_1, C_2, C_3, \ldots, C_n) M \begin{pmatrix} C_1 \\ C_2 \\ C_3 \\ \cdot \\ \cdot \\ \cdot \\ C_n \end{pmatrix}$$

$$- (0, 0, 0, \ldots 0, \overline{C}_{R+1}, \overline{C}_{R+2}, \ldots, \overline{C}_n) M \begin{pmatrix} C_1 \\ C_2 \\ \cdot \\ \cdot \\ \cdot \\ C_3 \end{pmatrix}$$

$$+ C_{SR} \left[\frac{1}{2} C_R + \frac{1}{4} C_{R+1} + \ldots \right] \tag{B.15}$$

The last two terms in Eq. (B.15) is always nonnegative. Moreover, the sum of these two terms is nondecreasing as R decreases. Since by Theorem A.1, $C_1 = 1$ is a condition for maximizing the first term of Eq. (B.15), it must also be a condition for maximizing Eq. (B.15) itself.

Eq. (B.11) may be rewritten with this condition as follows

$$-H = y_{-1} + (C_2 - C_{S2}, C_3 - C_{S3}, \ldots, C_n - C_{Sn}) M \begin{pmatrix} y_{-2} \\ \cdot \\ \cdot \\ \cdot \\ y_{-n} \end{pmatrix} \tag{B.16}$$

Therefore, the equation for H_{min} can be immediately written as:

$$H_{min}(n) = -1 - G_{max}(n-1) = -\frac{10}{9} - \frac{n-1}{3} - \frac{(-1)^n}{9.2^{n-1}} \tag{B.17}$$

Based on this analysis C_{S1}, x_{S1}, C_1, x_1, and y_{-1} are equal to 1 for H_{min}. The remaining stages are determined such as to maximize G for an n - 1 stage BRM. The values of y, C, and C_S where H_{min} is attained are listed in Table B.3.

TABLE B.3 C, y, AND C_S WHERE H_{min} IS ATTAINED

n	$y_{-1} \cdots y_{-6}$	$C_6 \cdots C_1$	$C_{S6} \cdots C_{S1}$
2	11	11	01
3	101	101	011
4	1011	1101	0011
5	10101	10101	01011
6	101011	110101	001011
n	$y_{-1} \cdots y_{-6}$	$C_6 \cdots C_1$	$C_{S6} \cdots C_{S1}$
2	11	11	01
3	111	111	001
4	1101	1011	0101
5	11011	11011	00101
6	110101	101011	010101

The BRM counter value and initial value for these H_{min} values are the 2's complement of the above numbers. These values are tabulated in Table 2.7.